INFORMATION AND COMPUTATION 117, 98–114 (1995)

# A Trace Semantics for Petri Nets*

P. W. HOOGERS AND H. C. M. KLEIJN

*Department of Computer Science, Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands*

AND

P. S. THIAGARAJAN

*School of Mathematics, SPIC Science Foundation, 92, G.N. Chetty Road, T. Nagar, Madras 600017, India*

A generalization of the notion of trace is proposed. This enables us to associate with each Petri net a single behavioural object, namely a poset of (generalized) traces. A characterization is given of the trace languages defined by Petri nets. We show that the general event structures of Winskel and the stable event structures can also be characterized in terms of our trace languages. One consequence is that in this framework, stable event structures, general event structures, and Petri nets constitute a strictly ascending chain in terms of expressive power. 

## INTRODUCTION

Our aim is to extend the semantic theory of elementary net systems [Nielsen *et al.*, 1990, 1992, to appear] to general Petri nets. We wish to provide a semantics for Petri nets which takes into account both the non-sequential and the branching aspects of their behaviour. Moreover, we would like to associate a *single* behavioural object with each Petri net. For elementary net systems, and in fact for 1-safe Petri nets, the classical notion of unfoldings based on labelled occurrence nets and the attendant notions of prime event structures and prime algebraic coherent domains can be used to achieve this purpose [Nielsen *et al.*, 1981; Nielsen *et al.*, 1990].

In the case of a Petri net it is far from clear what a proper notion of an unfolding is. The proposals that have been made in the literature are not very satisfactory in the sense that they more or less view tokens as "coloured" entities [Goltz and Mycroft, 1984; Goltz and Reisig, 1985; Engelfriet, 1991]. As a result, in these proposals, general Petri nets are—in terms of the event structures and domains that they yield—no more expressive than 1-safe Petri nets.

Conventional non-sequential processes (based on labelled causal nets) have also been proposed to capture the behaviour of Petri nets [Goltz and Reisig, 1983; Best and Devillers, 1987; Best and Fernández C., 1988]. This notion of non-sequential processes is tied to the intuition concerning the behaviour of 1-safe Petri nets and so once again tokens are treated as coloured entities. Some proposals have been made in order to identify processes which only differ in the colouring of the tokens [Best and Devillers, 1987; Vogler, 1990]. There is, however, very little hope of "sewing" together these processes to obtain a single behavioural object which takes into account the branching aspects of the behaviour, in contrast to the case of elementary net systems.

Another approach for describing the behaviour of elementary net systems and 1-safe Petri nets is based on trace theory [Mazurkiewicz, 1987; Rozenberg and Thiagarajan, 1986; Aalbersberg and Rozenberg, 1988]. Here the concurrency present in the net is formalized in a binary independence relation over transitions which induces an equivalence relation for the sequential runs of the net. Each of the resulting equivalence classes, which are called traces, represents a single non-sequential run. The behaviour of the net can then be described as a poset of traces ordered by a prefix ordering. In [Nielsen *et al.*, 1990] the close relationship between this approach and the methods mentioned above is established.

For (general) Petri nets, however, this approach does not work. The problem is that a place in a Petri net may contain many tokens. As a result concurrency (and conflict) are no longer global structural relations, but depend on the current marking. Moreover, a *multiset* of transitions may occur concurrently at a marking.

To deal with these phenomena, we generalize the theory of trace languages along three dimensions. First, we consider *step sequences* instead of ordinary sequences; by a step we mean here a finite multiset. Second, we consider

* An extended abstract (without proofs) of this paper was presented at ICALP'92 in Vienna and appeared in the proceedings of this conference in "Lecture Notes in Computer Science, Vol. 623" (W. Kuich, Ed.), pp. 595–604, Springer-Verlag, Berlin/New York, 1992.

0890-5401/95 $6.00

independence relations that are context-dependent, where the context is defined by a step sequence. Third, we specify in the independence relation a finite multiset of actions that can occur concurrently at a context rather than just a pair of symbols that may commute as in the classical case. It is then straightforward to lift the standard notions from the theory of trace languages to the—much richer—new setting.

The paper is organized as follows. In the next section, after some preliminaries, we introduce our generalized trace languages. In Section 2 we show how to obtain the trace behaviour of Petri nets using these generalized trace languages. Section 3 provides a characterization of the trace languages—called PN-trace languages—that are associated with the behaviour of Petri nets. In Section 4 we show that our trace languages are of some independent interest as a model of concurrency. In particular, we provide representations of general event structures [Winskel, 1987a] in terms of these trace languages. We then provide a characterization of the trace languages—called ES-trace languages—that arise in this fashion. One consequence of our results is that the class of ES-trace languages is strictly included in the class of PN-trace languages. In other words, Petri nets are more expressive than general event structures in this framework. We also characterize the trace languages associated with stable event structures, which "correspond" to the behaviour of 1-safe Petri nets [Winskel, 1987a]. Finally, in the concluding section, we briefly discuss the relationship between our generalized trace languages and the original trace model; we also discuss some related work.

## 1. GENERALIZED TRACE LANGUAGES

We start with some notations and conventions used throughout the paper.

All functions are total, unless explicitly stated otherwise. As usual, for a (partial) function $g: U_1 \to U_2$ and a subset $U_3 \subseteq U_1$, $g(U_3)$ is the set $\{g(a) \mid a \in U_3 \cap \text{domain}(g)\}$. A singleton set may be notationally identified with its only element.

Let $U$ be a set. The set of finite subsets of $U$ is denoted by $P_{\text{fin}}(U)$. A *multiset over* $U$ is a function $u: U \to \mathbb{N}$. For two multisets $u, v$ over $U$, their sum $u+v$ is the multiset defined by $(u+v)(a) = u(a) + v(a)$ for all $a \in U$; we write $v \leqslant u$ if $u = v + w$ for some multiset $w$, and if $v \leqslant u$, then $u - v$ is the (unique) multiset $w$ such that $u = v + w$. A multiset $u$ over $U$ with the property that $u(a) \leqslant 1$ for all $a \in U$ may be identified with the subset $\{a \in U \mid u(a) = 1\}$ of $U$. In particular, if $u$ is such that there is precisely one element $a \in U$ with $u(a) = 1$ and $u(b) = 0$ for all $b \in U$ with $b \neq a$, then we simply write $a$ for $u$. A multiset $u: U \to \mathbb{N}$ is *finite* if $\sum_{a \in U} u(a) \leqslant \infty$. The set of finite multisets over $U$ is denoted by $U^{\otimes}$. Note that the empty multiset $\underline{0}: U \to \mathbb{N}$ with $\underline{0}(a) = 0$, for all $a \in U$, is a member of $U^{\otimes}$. For $u \in U^{\otimes}$, $\#u = \sum_{a \in U} u(a)$ is the number of elements in $u$.

Let $\Sigma$ be an alphabet; that is, $\Sigma$ is a (possibly infinite) set. By $\Sigma^*$ we denote the free monoid generated by $\Sigma$. The product operation is concatenation and the elements of $\Sigma^*$ are called *words* or alternatively *sequences* (over $\Sigma$). The unit element of $\Sigma^*$ is the empty word $\Lambda$. Let $\Sigma^+ = \Sigma^* - \{\Lambda\}$ be the set of non-empty words over $\Sigma$.

Elements of $\Sigma^{\otimes}$ will be referred to as *steps* (*over* $\Sigma$). We let $\Sigma^{\#}$ denote the set of non-empty *step sequences* (over $\Sigma$). Thus $\Sigma^{\#} = (\Sigma^{\otimes})^+$. We view $\Sigma^{\#}$ as a (free) monoid: its unit element is $\underline{0} \in \Sigma^{\otimes}$; its product operation is the accordingly modified usual concatenation operation. Thus $\rho\underline{0} = \underline{0}\rho = \rho$ for all $\rho \in \Sigma^{\#}$, where $\rho\underline{0}$ denotes the product of $\rho$ and $\underline{0}$.

For $a \in \Sigma$ and $\rho \in \Sigma^{\#}$, we let $\text{num}_a(\rho)$ denote the number of times $a$ occurs in $\rho$. Thus $\text{num}_a(\underline{0}) = 0$ and $\text{num}_a(\rho u) = \text{num}_a(\rho) + u(a)$ for all $\rho \in \Sigma^{\#}$, $a \in \Sigma$, and $u \in \Sigma^{\otimes}$. For $\rho \in \Sigma^{\#}$ we let $\text{num}(\rho)$ denote the number of elements in $\rho$, that is, $\sum_{a \in \Sigma} \text{num}_a(\rho)$. For a step sequence $\rho \in \Sigma^{\#}$, $\text{alph}(\rho)$ denotes the set of elements from $\Sigma$ occurring in $\rho$; thus $\text{alph}(\rho) = \{a \in \Sigma \mid \text{num}_a(\rho) > 0\}$.

For a partial function $f: \Sigma_1 \to \Sigma_2$ with $\Sigma_1$ and $\Sigma_2$ alphabets, $\hat{f}: (\Sigma_1)^{\#} \to (\Sigma_2)^{\#}$ is the homomorphism induced by $f$ which is defined by $\hat{f}(u)(a) = \sum_{f(b)=a} u(b)$ if $a \in f(\Sigma_1)$ and $\hat{f}(u)(a) = 0$ otherwise. Note that $\hat{f}$ is total and that if $f: \Sigma_1 \to \Sigma_2$ is a bijection, then $\hat{f}: (\Sigma_1)^{\#} \to (\Sigma_2)^{\#}$ is also a bijection.

We generalize the classical notion of a trace language (see, e.g., [Mazurkiewicz, 1987; Aalbersberg and Rozenberg, 1988]) in essentially three ways. First, we consider step sequences rather than ordinary sequences. Second, we consider independence relations that are context-dependent. Finally, the independence relation specifies, for chosen contexts, a (*finite*) *multiset* of actions that can occur concurrently. The intuition motivating these three types of generalizations has been discussed in the Introduction and we will come back to it once we define the trace behaviour of Petri nets. For now, we quickly introduce the generalized trace languages we have in mind.

DEFINITION 1.1. A (*generalized*) *concurrency alphabet* is a pair $(\Sigma, \mathbf{I})$, where $\Sigma$ is an alphabet and $\mathbf{I} \subseteq \Sigma^{\#} \times \Sigma^{\otimes}$ is an *independence relation* (*over* $\Sigma$).

DEFINITION 1.2. Let $(\Sigma, \mathbf{I})$ be a concurrency alphabet.

(i) $\doteq_{\mathbf{I}} \subseteq \Sigma^{\#} \times \Sigma^{\#}$ is given thus: $\rho \doteq_{\mathbf{I}} \rho'$ iff there exist $\rho_1, \rho_2 \in \Sigma^{\#}$ and there exist $u, v, u', v' \in \Sigma^{\otimes}$ such that the following conditions are satisfied:

(i.a) $\rho = \rho_1 uv \rho_2$ and $\rho' = \rho_1 u'v'\rho_2$.

(i.b) $u + v = u' + v'$.

(i.c) $(\rho_1, u+v) \in \mathbf{I}$.

(ii) $\equiv_{\mathbf{I}} = (\doteq_{\mathbf{I}})^*$, where $(\doteq_{\mathbf{I}})^*$ denotes the reflexive transitive closure of $\doteq_{\mathbf{I}}$.

(iii) For $\rho \in \Sigma^*$, $[\rho]_{\mathbf{I}} = \{\rho' \in \Sigma^* \mid \rho \equiv_{\mathbf{I}} \rho'\}$ is the *trace* containing $\rho$.

(iv) $\Sigma^*/\equiv_{\mathbf{I}} = \{[\rho]_{\mathbf{I}} \mid \rho \in \Sigma^*\}$ is the *set of traces generated by* $(\Sigma, \mathbf{I})$.

(v) A (*generalized*) *trace language* (over $(\Sigma, \mathbf{I})$) is a subset of $\Sigma^*/\equiv_{\mathbf{I}}$.

If the concurrency alphabet $(\Sigma, \mathbf{I})$ is clear from the context we may omit the subscript $_{\mathbf{I}}$ in $\doteq_{\mathbf{I}}$, $\equiv_{\mathbf{I}}$ and $[\rho]_{\mathbf{I}}$.

Our definition is a generous one in that no restrictions have been imposed on the independence relation. In applications, one might wish to place some suitable restrictions on the independence relation to capture the intended interpretation. For instance, where the trace languages are used to model the behaviour of distributed systems one might demand:

(D1) $(\rho, u) \in \mathbf{I} \Rightarrow$ for all $v \leqslant u$: $(\rho, v) \in \mathbf{I}$ and

(D2) $(\rho, u) \in \mathbf{I} \Rightarrow$ for all $v \leqslant u$: $(\rho v, u - v) \in \mathbf{I}$.

(D1) and (D2) would capture the intuition that $(\rho, u) \in \mathbf{I}$ denotes the fact that the occurrences of actions mentioned in $u$ are independent of each other at the state represented by $\rho$.

A third reasonable axiom one could demand would be

(D3) $\rho \equiv \rho' \Rightarrow ((\rho, u) \in \mathbf{I}$ iff $(\rho', u) \in \mathbf{I})$.

This would ensure that $\mathbf{I}$ "agrees" with the equivalence relation $\equiv$ induced by it.

However, we will not require such restrictions at this stage. The subclasses of trace languages considered later in Sections 2, 3, and 4 will all have independence relations satisfying (D1), (D2), and (D3).

Let $(\Sigma, \mathbf{I})$ be a concurrency alphabet. Clearly, whenever $\rho, \rho' \in \Sigma^*$ are such that $\rho \equiv \rho'$, then $\rho\sigma \equiv \rho'\sigma$ for all $\sigma \in \Sigma^*$. Thus the usual "prefix" ordering over (classical) traces can be smoothly brought into the present framework, leading to the following well-defined notion.

The partial ordering $\leqslant_{\mathbf{I}} \subseteq (\Sigma^*/\equiv) \times (\Sigma^*/\equiv)$ is given by $[\rho] \leqslant_{\mathbf{I}} [\rho']$ iff there exists $\sigma \in \Sigma^*$ such that $\rho\sigma \equiv \rho'$. Clearly, $(\Sigma^*/\equiv, \leqslant_{\mathbf{I}})$ is a poset with $[\underline{0}] = \{\underline{0}\}$ as its least element. Again we may omit the index $\mathbf{I}$ in $\leqslant_{\mathbf{I}}$ whenever the concurrency alphabet $(\Sigma, \mathbf{I})$ is clear from the context.

Note that the obvious concatenation operation over traces given by $[\rho] \circ [\rho'] = [\rho\rho']$, as in the classical case, is not a well-defined operation, due to the context-dependent nature of the independence relation. Consider, for example, the concurrency alphabet $(\{a, b, c\}, \{(\underline{0}, \{b, c\})\})$; then $bc \equiv cb$, but $abc \equiv acb$ does not hold.

Finally, we wish to adopt a useful notation. We will often specify a trace language over $(\Sigma, \mathbf{I})$ on the basis of its underlying language in $\Sigma^*$. This language is *consistent with respect to* $(\Sigma, \mathbf{I})$.

DEFINITION 1.3. Let $(\Sigma, \mathbf{I})$ be a concurrency alphabet and let $L \subseteq \Sigma^*$. Then $L$ is consistent (with respect to $(\Sigma, \mathbf{I})$) if $[\rho] \subseteq L$ for all $\rho \in L$.

A trace language Tr over $(\Sigma, \mathbf{I})$ is represented uniquely by the triple $(L, \Sigma, \mathbf{I})$ with $L = \{\rho \mid [\rho] \in \mathrm{Tr}\}$. A triple $(L, \Sigma, \mathbf{I})$ with $L \subseteq \Sigma^*$ consistent with respect to $(\Sigma, \mathbf{I})$ represents the trace language $\{[\rho] \mid \rho \in L\}$.

We now define behaviour-preserving morphisms between trace languages. A trace morphism is behaviour-preserving in the sense that the underlying language is preserved and that independence is preserved.

DEFINITION 1.4. Let $(L_i, \Sigma_i, \mathbf{I}_i)$, $i = 1, 2$, be a pair of trace languages. A (*trace*) *morphism from* $(L_1, \Sigma_1, \mathbf{I}_1)$ *to* $(L_2, \Sigma_2, \mathbf{I}_2)$ is a partial function $f: \Sigma_1 \to \Sigma_2$ such that $\hat{f}(L_1) \subseteq L_2$ and $\hat{f}(\mathbf{I}_1) \subseteq \mathbf{I}_2$.

In the above definition the extension of $\hat{f}$ to $(\Sigma_1)^* \cup \mathbf{I}_1$ is also denoted by $\hat{f}$; on $\mathbf{I}_1$ it is defined by $\hat{f}((\rho, u)) = (\hat{f}(\rho), \hat{f}(u))$ for all $(\rho, u) \in \mathbf{I}_1$.

Note that concurrency must be preserved *locally*. Consider, e.g., the trace language $(L, \Sigma, \mathbf{I})$ with $\Sigma = \{a, b, c\}$, $L = \Sigma^*$ and $\mathbf{I} = \Sigma^* \times \Sigma^{\otimes}$, and the trace language $(L, \Sigma, \mathbf{I}')$ with $\mathbf{I}' = (\Sigma^* \times \Sigma^{\otimes}) - \{(\{a\}, \{b, c\})\}$. Although $\rho \equiv_{\mathbf{I}} \rho'$ iff $\rho \equiv_{\mathbf{I}'} \rho'$ for all $\rho, \rho' \in L$, the identity function on $\Sigma$ is not a trace morphism from $(L, \Sigma, \mathbf{I})$ to $(L, \Sigma, \mathbf{I}')$. Note that on the other hand this function is a trace morphism from $(L, \Sigma, \mathbf{I}')$ to $(L, \Sigma, \mathbf{I})$.

For a trace language $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$, $\mathrm{id}_{\mathrm{Tr}}: \Sigma \to \Sigma$ is the *identity trace morphism* from Tr to Tr defined by $\mathrm{id}_{\mathrm{Tr}}(a) = a$ for all $a \in \Sigma$. We may write id rather than $\mathrm{id}_{\mathrm{Tr}}$ whenever Tr is clear from the context.

In a standard way trace morphisms induce *trace isomorphisms*.

DEFINITION 1.5. Let $\mathrm{Tr}_i$, $i = 1, 2$, be a pair of trace languages. A trace morphism $f$ from $\mathrm{Tr}_1$ to $\mathrm{Tr}_2$ is a (*trace*) *isomorphism* if there exists a trace morphism $g$ from $\mathrm{Tr}_2$ to $\mathrm{Tr}_1$ such that $g \circ f = \mathrm{id}_{\mathrm{Tr}_1}$ and $f \circ g = \mathrm{id}_{\mathrm{Tr}_2}$. In that case $g$ is called the *inverse* of $f$.

It is a standard observation that the inverse of a trace isomorphism is unique. The following characterization of trace isomorphisms will be useful in later sections.

THEOREM 1.6. *Let* $\mathrm{Tr}_i = (L_i, \Sigma_i, \mathbf{I}_i)$, $i = 1, 2$, *be a pair of trace languages and let* $f: \Sigma_1 \to \Sigma_2$ *be a partial function. Then* $f$ *is a trace isomorphism from* $\mathrm{Tr}_1$ *to* $\mathrm{Tr}_2$ *iff* $f$ *is a bijection,* $\hat{f}(L_1) = L_2$, *and* $\hat{f}(\mathbf{I}_1) = \mathbf{I}_2$.

*Proof.* This follows easily from the definitions. ∎

If there exists a trace isomorphism from a trace language $\mathrm{Tr}_1$ to a trace language $\mathrm{Tr}_2$, then we say that $\mathrm{Tr}_1$ and $\mathrm{Tr}_2$ are (*trace*) *isomorphic* and we denote this by $\mathrm{Tr}_1 \approx \mathrm{Tr}_2$.

## 2. A TRACE SEMANTICS FOR PETRI NETS

Our aim in this section is to provide a semantics for Petri nets by associating a poset of generalized traces with every Petri net analogous to the approach followed for elementary net systems and 1-safe Petri nets in, e.g., [Mazurkiewicz, 1987].

For general nets however, things are much more complicated. As we mentioned in the Introduction, the current marking determines whether or not enabled transitions are independent, whereas for 1-safe Petri nets or elementary net systems independence of transitions is determined by the structure of the underlying net. This leads us to associate with a Petri net a context-dependent independence relation, specifying, given a history, which transitions are independent. A multiset of transitions is independent after a history if the marking to which this history leads provides each place of the net with enough input tokens for all occurrences of transitions mentioned in the multiset.

Now let us introduce formally the general Petri nets we have in mind. The Petri nets we consider have (finite) arc-weights; multiple occurrences of tokens in places are allowed. The transitions are not labelled.

DEFINITION 2.1. A *net (with arc-weights)* is a triple $(S, T, W)$ where

(i) $S$ is a set of *places*

(ii) $T$ is a set of *transitions*

(iii) $S \cap T = \varnothing$

(iv) $W: (S \times T) \cup (T \times S) \to \mathbb{N}$ is a *weight function.*

Thus a net may be viewed as a directed bipartite graph in which a weight greater than or equal to 1 is assigned to each edge.

The following convention for depicting nets is used. Places are drawn as circles and transitions as boxes. For $x, y \in S \cup T$, a directed arc is drawn from $x$ to $y$ iff $W(x, y) > 0$. This arc is labelled with its weight $W(x, y)$ if this weight is greater than 1.

A *marking* of a net is a multiset over its set of places. A marking $M$ of a net will be represented in the graphical representation of the net by drawing $M(s)$ tokens in each place $s$.

DEFINITION 2.2. A *Petri net* is a quadruple $N = (S, T, W, M_{\text{in}})$, where

(i) $(S, T, W)$ is a net

(ii) $M_{\text{in}} \in \mathbf{M}_N$ is the *initial marking of* $N$, where $\mathbf{M}_N$ denotes the set of all markings of $(S, T, W)$.

In order to formalize the dynamics of a Petri net, we associate a set of *step firing sequences* and a *step transition relation* with each Petri net, based on the usual firing rule for Petri nets.

DEFINITION 2.3. Let $N = (S, T, W, M_{\text{in}})$ be a Petri net. The set $SFS_N \subseteq T^{\#}$ of *step firing sequences* of $N$ and the *step transition relation* $\Rightarrow_N \subseteq \{M_{\text{in}}\} \times SFS_N \times \mathbf{M}_N$ are the least sets such that

(i) $\underline{0} \in SFS_N$ and $(M_{\text{in}}, \underline{0}, M_{\text{in}}) \in \Rightarrow_N$

(ii) if $\rho \in SFS_N$ and $(M_{\text{in}}, \rho, M) \in \Rightarrow_N$ and if $u \in T^{\otimes}$ is such that $M(s) \geq \sum_{t \in T} u(t) \cdot W(s, t)$ for all $s \in S$, then $\rho u \in SFS_N$ and $(M_{\text{in}}, \rho u, M') \in \Rightarrow_N$, where $M'(s) = M(s) + \sum_{t \in T} u(t) \cdot (W(t, s) - W(s, t))$ for all $s \in S$.

We write $M_{\text{in}} \stackrel{\rho}{\Longrightarrow}_N M$ rather than $(M_{\text{in}}, \rho, M) \in \Rightarrow_N$. Whenever $N$ is clear from the context we may denote $\mathbf{M}_N$ as $\mathbf{M}$, $\Rightarrow_N$ as $\Rightarrow$, and $SFS_N$ as $SFS$.

LEMMA 2.4. *Let $N = (S, T, W, M_{\text{in}})$ be a Petri net and let $\rho \in SFS$ and $M \in \mathbf{M}$ be such that $M_{\text{in}} \stackrel{\rho}{\Longrightarrow}_N M$. Then $M(s) = M_{\text{in}}(s) + \sum_{t \in T} \text{num}_t(\rho) \cdot (W(t, s) - W(s, t))$ for all $s \in S$.*

*Proof.* If $\rho = \underline{0}$ then the claim holds trivially. Now assume that $\rho = \rho' u$ with $\rho' \in SFS$ and $\underline{0} \neq u \in T^{\otimes}$. Let $s \in S$. By an inductive argument we assume that $M'(s) = M_{\text{in}}(s) + \sum_{t \in T} \text{num}_t(\rho') \cdot (W(t, s) - W(s, t))$ where $M' \in \mathbf{M}$ is such that $M_{\text{in}} \stackrel{\rho'}{\Longrightarrow} M'$. Now we can conclude that $M(s) = M'(s) + \sum_{t \in T} u(t) \cdot (W(t, s) - W(s, t)) = M_{\text{in}}(s) + \sum_{t \in T} \text{num}_t(\rho') \cdot (W(t, s) - W(s, t)) + \sum_{t \in T} u(t) \cdot (W(t, s) - W(s, t)) = M_{\text{in}}(s) + \sum_{t \in T} \text{num}_t(\rho) \cdot (W(t, s) - W(s, t))$. ∎

EXAMPLE 2.5. Let $N_1$, $N_2$, and $N_3$ be the Petri nets depicted in Fig. 1. In each of these three Petri nets transition $t$ can occur at most three times. The only step firing sequence of $N_1$ containing three occurrences of $t$ is $ttt$. The step firing sequences of $N_2$ containing three occurrences of $t$ are either $ttt$, or of the form $ut$ or $tu$, where $u$ is such that $u(t) = 2$; hence at most two "concurrent" occurrences of $t$ are possible. In addition to these step firing sequences, $N_3$ has also $v$ with $v(t) = 3$ as a step firing sequence with three occurrences of $t$.

We can now associate an independence relation $\mathbf{I}_N$ with a Petri net $N = (S, T, W, M_{\text{in}})$ in order to obtain a concurrency alphabet $(T, \mathbf{I}_N)$. This definition is based on the observation that steps in step firing sequences are multisets of "concurrently" enabled transitions; as mentioned earlier, we view such multisets as being independent.

DEFINITION 2.6. Let $N = (S, T, W, M_{\text{in}})$ be a Petri net. The *independence relation of* $N$ is the set $\mathbf{I}_N \subseteq T^{\#} \times T^{\otimes}$ with $\mathbf{I}_N = \{(\rho, u) \mid \rho u \in SFS \text{ with } \rho \in T^{\#} \text{ and } u \in T^{\otimes}\}$.

In order to simplify the notation we write $\doteq_N$, $\equiv_N$, $\leqslant_N$, and $[\rho]_N$ with $\rho \in T^{\#}$ instead of $\doteq_{\mathbf{I}_N}$, $\equiv_{\mathbf{I}_N}$, $\leqslant_{\mathbf{I}_N}$, and $[\rho]_{\mathbf{I}_N}$, respectively, in what follows.

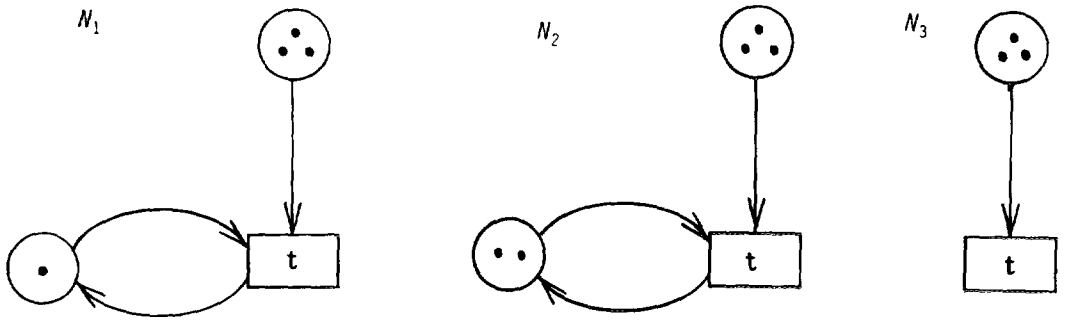


FIGURE 1

It is easy to verify that the independence relation $\mathbf{I}_N$ of a Petri net $N$ satisfies the properties

(D1) $(\rho, u) \in \mathbf{I}_N \Rightarrow$ for all $v \leqslant u$: $(\rho, v) \in \mathbf{I}_N$ and

(D2) $(\rho, u) \in \mathbf{I}_N \Rightarrow$ for all $v \leqslant u$: $(\rho v, u - v) \in \mathbf{I}_N$

mentioned in Section 1.

As a result any step in a step firing sequence can be split into arbitrary substeps. A consequence of this is that all traces determined by step firing sequences contain "sequential" representatives in which the steps are singletons.

From Lemma 2.4 it follows directly that step firing sequences which contain an equal number of occurrences of each transition lead to the same marking and thus determine the same independent multisets of transitions.

LEMMA 2.7. *Let $N = (S, T, W, M_{\text{in}})$ be a Petri net and let $\rho, \rho' \in SFS$ be such that $\text{num}_t(\rho) = \text{num}_t(\rho')$ for all $t \in T$ and let $u \in T^{\otimes}$. Then $\rho u \in SFS$ iff $\rho' u \in SFS$.*

As a consequence of the next lemma, every Petri net $N = (S, T, W, M_{\text{in}})$ generates a trace language $(SFS_N, T, \mathbf{I}_N)$.

LEMMA 2.8. *Let $N = (S, T, W, M_{\text{in}})$ be a Petri net. Then $SFS$ is consistent with respect to $(T, \mathbf{I}_N)$.*

*Proof.* Let $\rho \in SFS$ and $\rho' \in T^{\#}$ be such that $\rho \doteq_N \rho'$. Clearly, it is sufficient to prove that $\rho' \in SFS$. Let $\rho_1, \rho_2 \in T^{\#}$ and $u, v, u', v' \in T^{\otimes}$ be such that $\rho = \rho_1 u v \rho_2$, $\rho' = \rho_1 u' v' \rho_2$, $u + v = u' + v'$, and $(\rho_1, u + v) \in \mathbf{I}_N$. By the prefix-closedness of $SFS$ we have $\rho_1 uv \in SFS$. Furthermore $(\rho_1 u', v') \in \mathbf{I}_N$ because $\mathbf{I}_N$ satisfies the property (D2) mentioned above. Hence $\rho_1 u' v' \in SFS$ by the definition of $\mathbf{I}_N$. Since $\text{num}_t(\rho_1 uv) = \text{num}_t(\rho_1 u' v')$ for all $t \in T$, $\rho' \in SFS$ now follows from Lemma 2.7.

Lemma 2.7 and Lemma 2.8 imply that $\mathbf{I}_N$ also satisfies the property (D3) mentioned in Section 1:

(D3) $\rho \equiv_N \rho' \Rightarrow ((\rho, u) \in \mathbf{I}_N \text{ iff } (\rho', u) \in \mathbf{I}_N)$.

The trace language of a net induced by the set of its step firing sequences, together with the trace ordering, characterize the behaviour of a Petri net.

DEFINITION 2.9. Let $N = (S, T, W, M_{\text{in}})$ be a Petri net.

(i) $\mathbf{TR}_N = (SFS_N, T, \mathbf{I}_N)$ is the *trace language generated by N*.

(ii) The poset $(\mathbf{TR}_N, \leqslant_N)$ is the *trace behaviour of N*.

Note that in (ii) by a slight abuse of notation we have denoted the restriction of $\leqslant_N$ to $\mathbf{TR}_N \times \mathbf{TR}_N$ also as $\leqslant_N$. As usual we omit the subscript whenever possible and write $\mathbf{TR}$ rather than $\mathbf{TR}_N$.

EXAMPLE 2.5 (Continued). The trace behaviours of the Petri nets $N_1$, $N_2$, and $N_3$ are depicted in Fig. 2 in the form of Hasse diagrams; we show for each trace a sequential representative. The underlying posets of these trace behaviours are the same, even though the corresponding trace languages are not isomorphic. In order to investigate the concurrent aspects of the behaviours of these nets, one has to take the internal structure of the traces into account. We have, e.g., $[ttt]_{N_1} = \{ttt\}$, whereas $[ttt]_{N_2} = \{ttt, ut, tu\}$ with $u$ such that $u(t) = 2$ and $[ttt]_{N_3} = [ttt]_{N_2} \cup \{v\}$ with $v$ such that $v(t) = 3$.

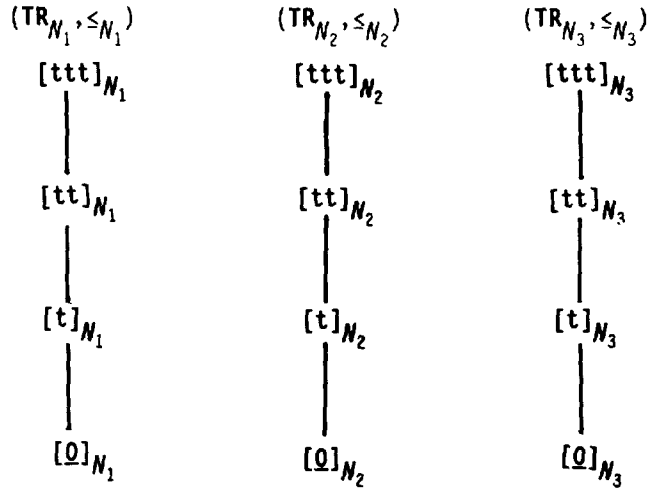


FIGURE 2

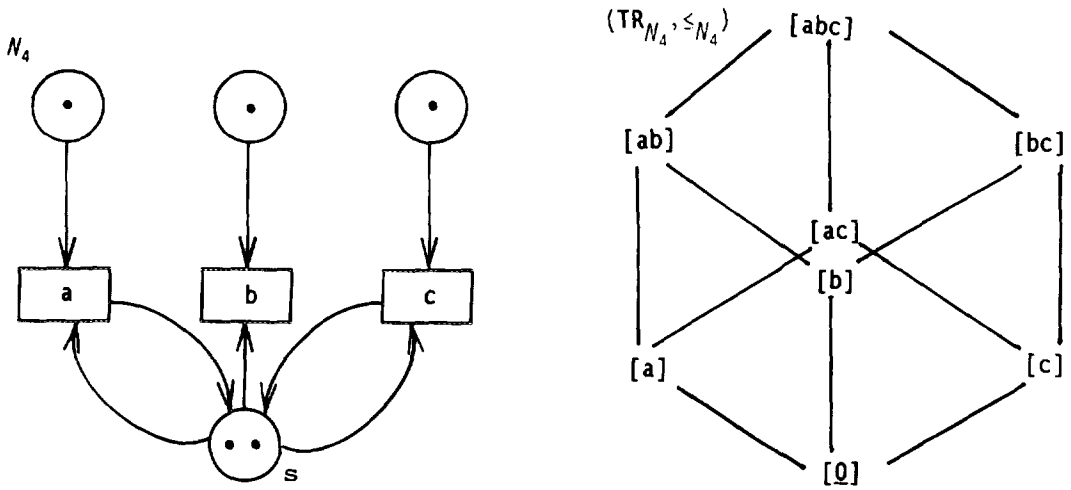


FIGURE 3

EXAMPLE 2.10. Let $N_4$ be the Petri net in Fig. 3 with its associated trace behaviour. In the trace behaviour of $N_4$ the trace $[ac]$ consists of the step firing sequences $ac$, $ca$, and $\{a, c\}$. The trace $[abc]$ contains the step firing sequences $bac$ and $bca$, but not the step sequence $b\{a, c\}$. Hence the history of the Petri net determines if $a$ and $c$ are independent.

Let $N_5$ be the Petri net in Fig. 4. In $N_5$ both $a$ and $c$ are step firing sequences, but there are not enough tokens for the step $\{a, c\}$ in the given marking. But after the occurrence of $b$ the step $\{a, c\}$ can occur. Hence once again the independence of $a$ and $c$ depends on the history of the Petri net.

Finally, let $N_6$ be the Petri net in Fig. 5. In the trace behaviour of $N_6$ the traces $[a]$ and $[c]$ have upperbounds $[ac]$ and $[abc]$, but they have no least upperbound. This means that $(\mathbf{TR}_{N_6}, \leqslant_{N_6})$ is not consistently complete.

We will return to some of these trace languages in the following sections.

## 3. PETRI NETS AND GENERALIZED TRACE LANGUAGES

In the previous section we have associated with every Petri net a trace language. Our aim in this section is to characterize those trace languages, called *PN-trace languages*, which are isomorphic to a trace language associated with a Petri net. To this aim we propose four axioms for trace languages. It is fairly easy to prove that

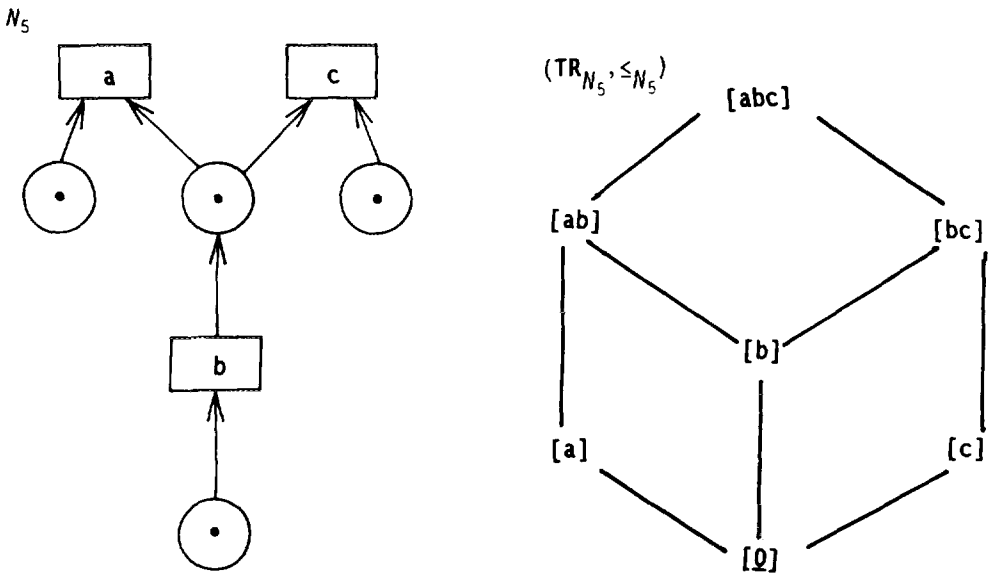


FIGURE 4

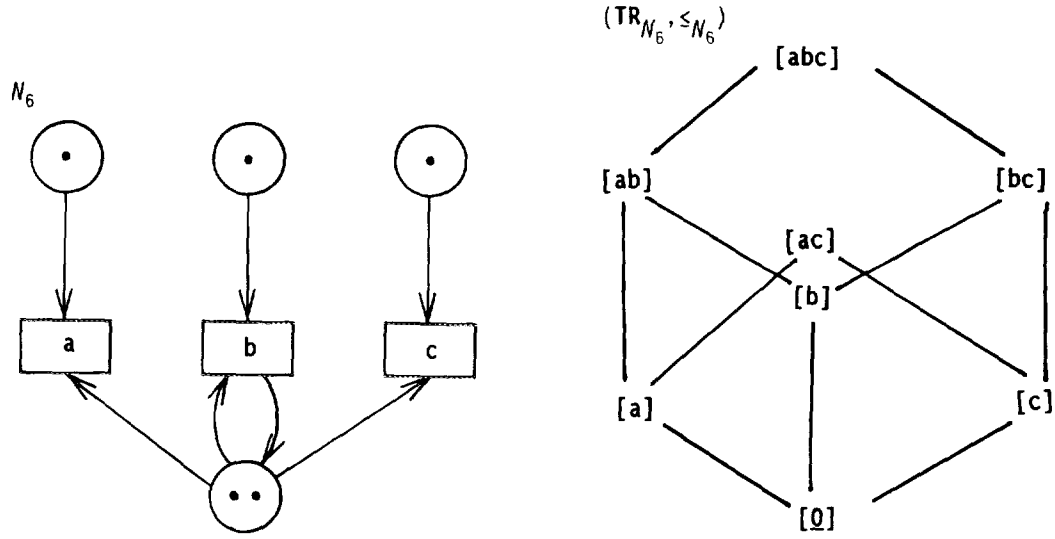


FIGURE 5

every PN-trace language satisfies these four axioms. In order to prove the converse, we associate a Petri net with every trace language satisfying the four axioms. We prove that the trace language generated by this Petri net and the original trace language are isomorphic.

DEFINITION 3.1. A trace language Tr is a *PN-trace language* if there exists a Petri net $N$ such that $\mathrm{Tr} \approx \mathbf{TR}_N$.

In order to formulate one of our axioms for characterizing the PN-trace languages, we need the notion of a *region* of a trace language Tr, which corresponds to the notion of a place of a Petri net. The regions discussed here generalize the notion of a region from [Ehrenfeucht and Rozenberg, 1990]. Recently a closely related idea has been exploited in [Mukund, 1992] to characterize the operational behaviour of Petri nets in terms of transition systems.

The definition of a region is based on a transition relation for trace languages.

DEFINITION 3.2. Let $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ be a trace language. The transition relation $\rightarrow_{\mathrm{Tr}} \subseteq L \times \Sigma^{\otimes} \times L$ is given by $\rightarrow_{\mathrm{Tr}} = \{(\rho, u, \rho') \mid \rho, \rho' \in L \text{ and } u \in \Sigma^{\otimes} \text{ such that } \rho u \equiv \rho'\}$.

In what follows we write $\rho \xrightarrow{u}_{\mathrm{Tr}} \rho'$ instead of $(\rho, u, \rho') \in \rightarrow_{\mathrm{Tr}}$. If the trace language Tr is clear from the context we may omit the subscript Tr from $\rightarrow_{\mathrm{Tr}}$.

Given a trace language and its associated transition relation, (generalized) regions are defined to model "local" states. They correspond to the places of a Petri net, and their definition is based on the following observation. Given a place $s$ of a Petri net $N$, as far as the effect of $s$ on the behaviour of $N$ at a marking $M$ is concerned, we only need to know $M(s)$ and, for each transition $t$, the pair of natural numbers $(W(s, t), W(t, s))$. Hence a place $s$ of $N$ may be viewed as a function $r_s: \mathbf{RM}_N \cup T \rightarrow \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$, where $\mathbf{RM}_N$ denotes the set of reachable markings of $N$. In fact, $r_s: SFS \cup T \rightarrow \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$. Consider, e.g., the place $s$ of the Petri net $N_4$ from Example 2.10. Then $s$ corresponds to the function $r_s$ with, for all $\rho \in SFS_{N_4}$, $r_s(\rho) = 2$ if $b \notin \mathrm{alph}(\rho)$ and $r_s(\rho) = 1$ otherwise, and $r_s(a) = r_s(c) = (1, 1)$ and $r_s(b) = (1, 0)$. Because the alphabet of a trace language corresponds to the set of transitions of a Petri net and the underlying language of a trace language corresponds to the set of step firing sequences of a Petri net, a region of a trace language $(L, \Sigma, \mathbf{I})$ is a function $r: L \cup \Sigma \rightarrow \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$. Because each place of a Petri net obeys the firing rule, we also demand that a region of a trace language satisfies the corresponding conditions.

DEFINITION 3.3. Let $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ be a trace language. A (*generalized*) *region* of Tr is a function $r: L \cup \Sigma \rightarrow \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$ satisfying the following conditions.

(i) For all $\rho \in L$: $r(\rho) \in \mathbb{N}$ and for all $a \in \Sigma$: $r(a) \in \mathbb{N} \times \mathbb{N}$.

(ii) If $\rho \xrightarrow{u} \rho'$ then $r(\rho) \geqslant \sum_{a \in \Sigma} u(a) \cdot {}^{r}a$ and $r(\rho') = r(\rho) + \sum_{a \in \Sigma} u(a) \cdot (a^{r} - {}^{r}a)$ where $r(a) = ({}^{r}a, a^{r})$ for all $a \in \Sigma$.

A region $r$ of Tr is said to be *non-trivial* iff $r(a) \neq (0, 0)$ for some $a \in \Sigma$. $\mathbf{R}_{\mathrm{Tr}}$ is the set of non-trivial regions of Tr.

Now we are ready to present our axioms. These axioms have been stated in terms of a trace language $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$. In stating these axioms and through the rest of the paper, we let $\rho$, $\rho'$, $\rho_1$, $\rho_2$, $\sigma$, etc. range over $\Sigma^{*}$ and $u$, $v$, $u'$, $v'$, etc., range over $\Sigma^{\otimes}$.

(PN1) $L \neq \varnothing$ (Non-emptiness).

(PN2) $\rho u \in L \Rightarrow \rho \in L$ (Prefix-closedness).

(PN3) $pu \in L \Leftrightarrow (p, u) \in \mathbf{I}$ (Substep property).

(PN4) $p \in L \Rightarrow (\forall r \in \mathbf{R}_{\mathrm{Tr}}\colon\ r(p) \geqslant \sum_{a \in \Sigma} u(a) \cdot {}^{r}a \Rightarrow pu \in L)$ (Forward closure property).

Now we turn to the proof that every PN-trace language satisfies these axioms. First of all we prove in the following lemma that they are preserved under isomorphisms.

LEMMA 3.4. *Let* $\mathrm{Tr}_i = (L_i, \Sigma_i, \mathbf{I}_i)$, $i = 1, 2$, *be a pair of trace languages such that* $\mathrm{Tr}_1 \approx \mathrm{Tr}_2$. *Then, for each* $i = 1, 2, 3, 4$, $\mathrm{Tr}_1$ *satisfies* (PNi) *iff* $\mathrm{Tr}_2$ *satisfies* (PNi).

*Proof.* Preservation of (PN1), (PN2), and (PN3) follows immediately from the characterization of trace isomorphisms given in Theorem 1.6.

In order to prove that $\mathrm{Tr}_2$ satisfies (PN4) if $\mathrm{Tr}_1$ does, let $f$ be an isomorphism from $\mathrm{Tr}_2$ to $\mathrm{Tr}_1$. Let $p \in L_2$ and $u \in (\Sigma_2)^{\otimes}$ be such that $pu \notin L_2$. Then we also have $\hat{f}(p) \in L_1$ and $\hat{f}(pu) \notin L_1$ by Theorem 1.6. Hence there exists $r \in \mathbf{R}_{\mathrm{Tr}_1}$ such that $r(\hat{f}(p)) < \sum_{a \in \Sigma_1} (\hat{f}(u))(a) \cdot {}^{r}a$ by (PN4). Define the function $r'\colon L_2 \cup \Sigma_2 \to \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$ by $r'(p') = r(\hat{f}(p'))$, for all $p' \in L_2$, and $r'(a) = r(f(a))$, for all $a \in \Sigma_2$. Because $\sum_{a \in \Sigma_2} u(a) \cdot {}^{r}a = \sum_{b \in \Sigma_1} (\hat{f}(u))(b) \cdot {}^{r}b$, it is easy to see that $r'$ is a non-trivial region of $L_2$. Moreover $r'(p) < \sum_{a \in \Sigma_2} u(a) \cdot {}^{r}a$. Hence $(L_2, \Sigma_2, \mathbf{I}_2)$ satisfies (PN4). Since $\approx$ is symmetric, this completes our proof. ∎

In order to show that every PN-trace language satisfies (PN1) through (PN4), we now only have to prove that every trace language generated by a Petri net satisfies these four axioms.

As a first step in proving this, the following lemma shows that every place s of a Petri net $N = (S, T, W, M_{\mathrm{in}})$ determines a region $r\langle s\rangle$ of $\mathbf{TR} = (SFS, T, \mathbf{I}_N)$ in the following way. For each $t \in T$, the function $r\langle s\rangle$ gives the number of tokens $t$ takes from $s$ and the number of tokens $t$ puts in $s$ and, for each $p \in SFS$, the function $r\langle s\rangle$ gives the number of tokens in $s$ after the step firing sequence $p$.

LEMMA 3.5. *Let* $N = (S, T, W, M_{\mathrm{in}})$ *be a Petri net and let* $s \in S$. *Define* $r\langle s\rangle\colon SFS \cup T \to \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$ *such that*

(i) $r\langle s\rangle(t) = (W(s, t), W(t, s))$ *for all* $t \in T$

(ii) $r\langle s\rangle(p) = M(s)$ *for all* $p \in SFS$ *where* $M \in \mathbf{M}$ *is such that* $M_{\mathrm{in}} \overset{p}{\Longrightarrow} \mathbf{M}$. *Then* $r\langle s\rangle$ *is a region of* **TR**.

*Proof.* Let $p, p' \in SFS$ and $u \in T^{\otimes}$ be such that $p \overset{u}{\longrightarrow} p'$. Then $p' \equiv pu$ and by the consistency of $SFS$ we have that $pu \in SFS$. Let $M, M'$ be the markings such that $M_{\mathrm{in}} \overset{p}{\Longrightarrow} \mathbf{M}$ and $M_{\mathrm{in}} \overset{pu}{\Longrightarrow} M'$. Then $r\langle s\rangle(p) = M(s) \geqslant \sum_{t \in T} u(t) \cdot W(s, t) = \sum_{t \in T} u(t) \cdot {}^{r\langle s\rangle}t$ and $r\langle s\rangle(p') = M'(s) = M(s) + \sum_{t \in T} u(t) \cdot (W(t, s) - W(s, t)) = r\langle s\rangle(p) + \sum_{t \in T} u(t) \cdot (t^{r\langle s\rangle} - {}^{r\langle s\rangle}t)$. ∎

Using the above lemma we can prove the following result.

LEMMA 3.6. *Let* $N$ *be a Petri net. Then* **TR** *satisfies the axioms* (PN1) *through* (PN4).

*Proof.* From the definition of $SFS$ and $\mathbf{I}_N$ it follows directly that **TR** satisfies (PN1), (PN2), and (PN3).

In order to prove that $(SFS, T, \mathbf{I}_N)$ satisfies (PN4), let $p \in SFS$ and $u \in T^{\otimes}$ be such that $pu \notin SFS$. Then by the definition of $SFS$ there exists $s \in S$ such that $M(s) < \sum_{t \in T} u(t) \cdot W(s, t)$ where $M \in \mathbf{M}$ is such that $M_{\mathrm{in}} \overset{p}{\Longrightarrow} M$. Consider the region $r\langle s\rangle$ from Lemma 3.5. Then $r\langle s\rangle(p) = M(s) < \sum_{t \in T} u(t) \cdot W(s, t) = \sum_{t \in T} u(t) \cdot {}^{r\langle s\rangle}t$. Since $r\langle s\rangle$ is non-trivial this implies that $(SFS, T, \mathbf{I}_N)$ satisfies (PN4). ∎

Lemma 3.6, together with Lemma 3.4, shows that all PN-trace languages satisfy (PN1) through (PN4). The remainder of this section is devoted to the proof that any trace language which satisfies the axioms (PN1) through (PN4) is a PN-trace language. Given a trace language $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ satisfying these four axioms, we show how to construct a Petri net $N_{\mathrm{Tr}}$ with the non-trivial regions of Tr as its places and $\Sigma$ as its transitions; Tr and the trace language generated by $N_{\mathrm{Tr}}$ are isomorphic, where the required isomorphism is the identity trace morphism.

DEFINITION 3.7. Let $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ be a trace language with $\mathbf{R}_{\mathrm{Tr}} \cap \Sigma = \varnothing$ satisfying (PN1) through (PN4). The *Petri net associated with* Tr is the Petri net $N_{\mathrm{Tr}} = (S, T, W, M_{\mathrm{in}})$ where $S = \mathbf{R}_{\mathrm{Tr}}$, $T = \Sigma$, $W\colon (\mathbf{R}_{\mathrm{Tr}} \times \Sigma) \cup (\Sigma \times \mathbf{R}_{\mathrm{Tr}}) \to \mathbb{N}$ is given by: $W(r, a) = {}^{r}a$ and $W(a, r) = a^{r}$ for all $r \in \mathbf{R}_{\mathrm{Tr}}$ and $a \in \Sigma$; $M_{\mathrm{in}}\colon \mathbf{R}_{\mathrm{Tr}} \to \mathbb{N}$ is given by $M_{\mathrm{in}}(r) = r(\underline{0})$ for all $r \in \mathbf{R}_{\mathrm{Tr}}$.

Note that $M_{\mathrm{in}}$ in the above definition is well defined because $\underline{0} \in L$ by (PN1) and (PN2).

In the rest of this paper we assume that the trace languages $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ we work with satisfy $\mathbf{R}_{\mathrm{Tr}} \cap \Sigma = \varnothing$. We can do this without loss of generality, because otherwise we could work with a Petri net obtained by indexing every place $r \in \mathbf{R}_{\mathrm{Tr}}$ in an appropriate way. For this new Petri net similar results can be derived. We will ignore this possible notational complication here.

EXAMPLE 3.8. Consider the Petri net $N_4$ from Example 2.10. By Lemma 3.6, $\mathbf{TR}_{N_4}$ satisfies the axioms (PN1) through (PN4). Every region $r$ of this trace language satisfies, e.g., the condition that $r(\underline{0}) > {}^{r}a + {}^{r}b$, because $\underline{0} \overset{\{a, b\}}{\longrightarrow} ab$.

Some regions of $\mathbf{TR}_{N_4}$ are:

for every place $s$ of $N_4$ its corresponding region $r\langle s\rangle$ as defined in Lemma 3.5;

$r_1$ with $r_1(a) = r_1(b) = (0, 0)$, $r_1(c) = (0, 3)$, and $r_1(p) = 3 * \mathrm{num}_c(p)$ for all $p \in SFS$;

$r_2$ with $r_2(a) = (0, 1)$, $r_2(b) = (1, 0)$, $r_2(c) = (0, 0)$, and $r_2(p) = 2 + \mathrm{num}_a(p) - \mathrm{num}_b(p)$ for all $p \in SFS$;

$r_{a,k}$ for $k > 0$ with $r_{a,k}(a) = (0, 1)$, $r_{a,k}(b) = r_{a,k}(c) = (0, 0)$, and $r_{a,k}(p) = k + \mathrm{num}_a(p)$ for all $p \in SFS$.

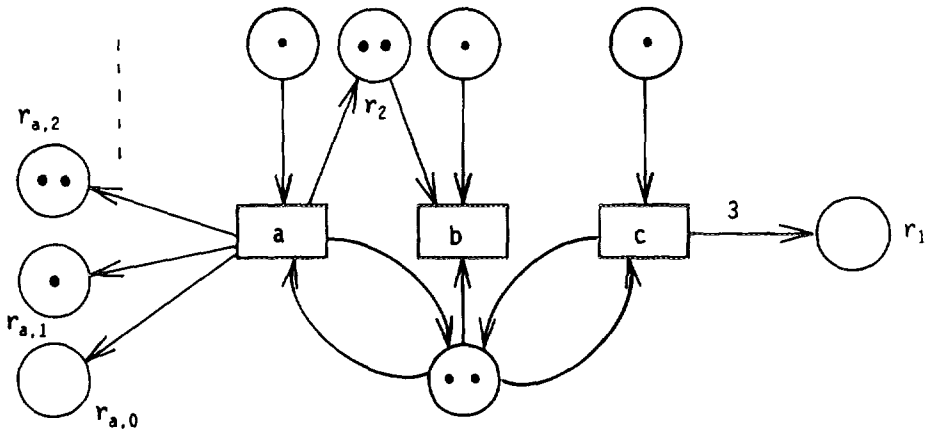


FIGURE 6

These regions form some (but not all) of the (infinite number of) places of the Petri net associated with this trace language and they are shown in Fig. 6.

As an intermediate step in the proof that a trace language $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ satisfying the axioms (PN1) through (PN4) is isomorphic to the trace language $\mathbf{TR}_{N_{\mathrm{Tr}}}$ generated by the Petri net $N_{\mathrm{Tr}}$, we prove separately that $L = SFS_{N_{\mathrm{Tr}}}$. To this aim we first show that for every place $s$ of $N_{\mathrm{Tr}}$, the region $r\langle s\rangle$ of $\mathbf{TR}_{N_{\mathrm{Tr}}}$ associated with $s$ in the way described in Lemma 3.5 and the region $s$ of Tr agree on all elements that their domains have in common. Thus, after it has been shown that $L = SFS_{N_{\mathrm{Tr}}}$, it follows that the non-trivial regions of $\mathbf{TR}_{N_{\mathrm{Tr}}}$ are precisely the places of $N_{\mathrm{Tr}}$.

LEMMA 3.9. *Let* $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1) *through* (PN4) *and let* $N_{\mathrm{Tr}} = (\mathbf{R}_{\mathrm{Tr}}, \Sigma, W, M_{\mathrm{in}})$ *be the Petri net associated with it. Then* $r\langle s\rangle(x) = s(x)$, *for each* $s \in \mathbf{R}_{\mathrm{Tr}}$ *and* $x \in \Sigma \cup (L \cap SFS_{N_{\mathrm{Tr}}})$, *where* $r\langle s\rangle$ *is defined as in Lemma* 3.5.

*Proof.* Let $s \in \mathbf{R}_{\mathrm{Tr}}$. By the definition in Lemma 3.5, we have that for each $a \in \Sigma$, $r\langle s\rangle(a) = (W(s, a), W(a, s))$ and this equals $s(a)$ according to Definition 3.7. Similarly, $r\langle s\rangle(\underline{0}) = M_{\mathrm{in}}(s)$ by Lemma 3.5 and $s(\underline{0}) = M_{\mathrm{in}}(s)$ by Definition 3.7. Note that $\underline{0} \in L \cap SFS$. Now let us assume that $r\langle s\rangle(\rho) = s(\rho)$ for all $\rho \in L \cap SFS$ with $\mathrm{num}(\rho) < n$ where $n \geqslant 1$. Let $\rho \in L \cap SFS$ be such that $\rho = \rho' u$ where $\rho \in \Sigma^{\#}$ and $u \in \Sigma^{\otimes}$ are such that $\mathrm{num}(\rho') < n$ and $\mathrm{num}(\rho' u) \geqslant n$. Note that $\rho' \xrightarrow{u}_{\mathrm{Tr}_{N_{\mathrm{Tr}}}} \rho$ and $\rho' \xrightarrow{u}_{\mathrm{Tr}} \rho$. Then $r\langle s\rangle(\rho) = M_{\mathrm{in}}(s) + \sum_{a \in \Sigma} \mathrm{num}_a(\rho) \cdot (W(a, s) - W(s, a))$ by Lemma 2.4. Thus $r\langle s\rangle(\rho) = M_{\mathrm{in}}(s) + \sum_{a \in \Sigma} \mathrm{num}_a(\rho') \cdot (W(a, s) - W(s, a)) + \sum_{a \in \Sigma} u(a) \cdot (W(a, s) - W(s, a)) = r\langle s\rangle(\rho') + \sum_{a \in \Sigma} u(a) \cdot (W(a, s) - W(s, a)) = s(\rho') + \sum_{a \in \Sigma} u(a) \cdot (W(a, s) - W(s, a))$ by the induction hypothesis and this in turn equals $s(\rho') + \sum_{a \in \Sigma} u(a) \cdot (a^s - {}^s a) = s(\rho)$ because $s$ is a region of $L$. Consequently $r\langle s\rangle(a) = s(a)$ for all $a \in \Sigma$ and $r\langle s\rangle(\rho) = s(\rho)$ for all $\rho \in L \cap SFS$. ∎

Using the correspondence described in this lemma between the regions of the original trace language and the regions of the trace language generated by the Petri net associated with this trace language, we can prove the following result.

LEMMA 3.10. *Let* $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1) *through* (PN4). *Then* $L = SFS_{N_{\mathrm{Tr}}}$.

*Proof.* In order to prove that $L \subseteq SFS_{N_{\mathrm{Tr}}}$, let $\rho \in L$. If $\rho = \underline{0}$, then $\rho \in SFS_{N_{\mathrm{Tr}}}$. If $\rho = \rho' u$ for some $\rho' \in \Sigma^{\#}$ and $u \in \Sigma^{\otimes}$ with $u \neq \underline{0}$, then $\rho' \in L$ by the prefix-closedness of $L$. Using an inductive argument we assume that $\rho' \in SFS_{N_{\mathrm{Tr}}}$. Let $M'$ be the marking such that $M_{\mathrm{in}} \xrightarrow{\rho'} M'$. By Lemma 3.9 we know that $M'(s) = r\langle s\rangle(\rho') = s(\rho')$ for each $s \in \mathbf{R}_{\mathrm{Tr}}$. Since $\rho' \xrightarrow{u}_{\mathrm{Tr}} \rho$ and $s$ is a region of Tr, it follows that $M'(s) = s(\rho') \geqslant \sum_{a \in \Sigma} u(a) \cdot {}^s a = \sum_{a \in \Sigma} u(a) \cdot W(s, a)$. Hence $\rho = \rho' u \in SFS$.

In order to prove that $SFS_{N_{\mathrm{Tr}}} \subseteq L$, let $\rho \in SFS_{N_{\mathrm{Tr}}}$. If $\rho = \underline{0}$, then $\rho \in L$ by the non-emptiness and prefix-closedness of $L$. Now assume that $\rho = \rho' u$ for some $\underline{0} \neq u \in (\Sigma_{N_{\mathrm{Tr}}})^{\otimes}$ and $\rho' \in (\Sigma_{N_{\mathrm{Tr}}})^{\#}$. Using an inductive argument we assume that $\rho' \in L$. Moreover, by Lemma 3.9, $s(\rho') = r\langle s\rangle(\rho') = M'(s)$ for all $s \in \mathbf{R}_{\mathrm{Tr}}$, where $M'$ is the marking such that $M_{\mathrm{in}} \xrightarrow{\rho'} M'$ and with $M_{\mathrm{in}}$ the initial marking of $N_{\mathrm{Tr}}$. Now assume that $\rho = \rho' u \notin L$. Then by (PN4), there exists a region $r \in \mathbf{R}_{\mathrm{Tr}}$ such that $r(\rho') < \sum_{a \in \Sigma} u(a) \cdot {}^r a$. This would imply that $M'(r) < \sum_{a \in \Sigma} u(a) \cdot W(r, a)$. But this contradicts the fact that $\rho' u \in SFS_{N_{\mathrm{Tr}}}$. Hence $\rho' u \in L$. ∎

We now have that the underlying language of the original trace language and the set of step firing sequences of the Petri net associated with this trace language are equal. In order to prove that the original trace language and the trace language generated by this Petri net are isomorphic, the only thing left to prove is that concurrency is preserved.

LEMMA 3.11. *Let* $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1) *through* (PN4). *Then* $\mathrm{Tr} \approx \mathbf{TR}_{N_{\mathrm{Tr}}}$.

*Proof.* We prove that the identity function on $\Sigma$ is an isomorphism from Tr to $\mathbf{TR}_{N_{\mathrm{Tr}}}$. According to Theorem 1.6 and Lemma 3.10 it is sufficient to prove that $\mathbf{I} = \mathbf{I}_{N_{\mathrm{Tr}}}$. This, however, follows immediately from Lemma 3.10, the substep property (PN3), and the definition of $\mathbf{I}_{N_{\mathrm{Tr}}}$. ∎

From Lemmas 3.4, 3.6, and 3.11 we obtain the following result.

THEOREM 3.12. *A trace language is a PN-trace language iff it satisfies the axioms* (PN1) *through* (PN4).

## 4. EVENT STRUCTURES AND GENERALIZED TRACE LANGUAGES

In this section we show that our generalized trace languages can also be used to capture yet another well-known model of concurrency known as *event structures*. Actually the term event structures is used in various senses; here we shall consider the general event structures specified as a family of configurations by Winskel in [Winskel, 1987a]. Since we do not deal with infinite step sequences it will be appropriate to consider only families of finite configurations. This, however, causes no loss of information due to the axiom of finite causes imposed on families of configurations in [Winskel, 1987a]. From now on we will refer to general event structures just as event structures.

After some general observations about event structures we associate a generalized trace language with every event structure. Analogously to the situation for PN-trace languages, we then define *ES-trace languages* as those trace languages which are isomorphic to a trace language associated with an event structure.

Our aim in the first part of this section is to characterize ES-trace languages. It is proved that ES-trace languages are precisely those trace languages satisfying certain axioms. In proving this result we associate an event structure with every trace language satisfying these axioms. The trace language associated with this event structure and the original trace language are isomorphic, where the isomorphism is the identity function.

We prove that the axioms we propose for characterizing ES-trace languages strengthen the axioms characterizing the PN-trace languages and so every ES-trace language is a PN-trace language. As a consequence, every event structure can be represented by a Petri net yielding the same trace language.

In the second part of this section we concentrate on the so-called *stable event structures*. It turns out that we can characterize the trace languages associated with stable event structures by adding one more axiom.

Let us now formally introduce the notion of event structure.

DEFINITION 4.1. An *event structure* is a pair $W = (E, C)$, where $E$ is a set of *events* and $\emptyset \neq C \subseteq P_{\mathrm{fin}}(E)$ is a family of (*finite*) *configurations* which satisfies the following two requirements.

(W1) For all $c \in C$: if $c \neq \emptyset$, then there exists $e \in c$ such that $c - \{e\} \in C$.

(W2) For all $c, c' \in C$: if $c \uparrow c'$ then $c \cup c' \in C$ (where $c \uparrow c'$ iff there exists $c'' \in C$ such that $c \subseteq c''$ and $c' \subseteq c''$).

Event structures satisfy the following "coincidence-freeness" property from [Winskel, 1987a].

LEMMA 4.2. *Let* $W = (E, C)$ *be an event structure and let* $c \in C$ *and* $e_1, e_2 \in c$ *be such that* $e_1 \neq e_2$. *Then there exists* $c' \in C$ *with* $c' \subseteq c$ *such that* $e_1 \in c'$ *iff* $e_2 \notin c'$.

*Proof.* If $c = \{e_1, e_2\}$ then we get from (W1) that $\{e_1\} \in C$ or $\{e_2\} \in C$. In either case the required result follows. If $c - \{e_1, e_2\} \neq \emptyset$ then, again by (W1), there exists $e \in c$ such that $c - \{e\} \in C$. If $e = e_1$ or $e = e_2$ then we can get $c - \{e\} = c'$. Otherwise by an inductive argument applied to $c - \{e\}$ the required $c' \in C$ exists. ∎

Here is an example of an event structure.

EXAMPLE 4.3. Let $W = (E, C)$ be the event structure with $E = \{a, b, c\}$ and $C = \{\emptyset, \{a\}, \{b\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$.

$W$ is depicted in Fig. 7, where the configurations are ordered under inclusion.

Concurrency within event structures is expressed through the following notion.

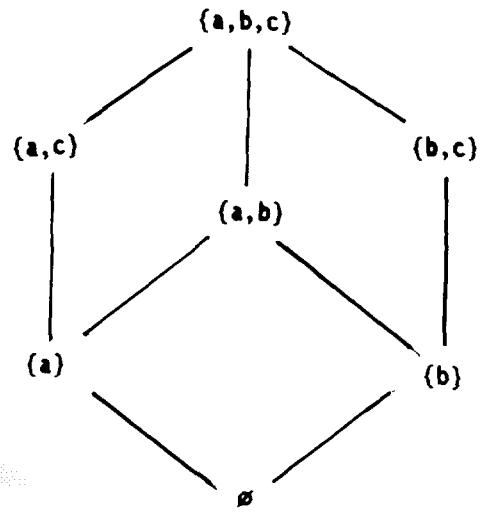


FIGURE 7

DEFINITION 4.4. Let $W=(E, C)$ be an event structure. Let $c \in C$ and $u \in P_{\text{fin}}(E)$. Then $u$ is *enabled* at $c$, denoted by $c\,[u\rangle_W$, if $c \cap u = \emptyset$ and $c \cup v \in C$ for all $v \subseteq u$.

If $W$ is clear from the context we may write $c\,[u\rangle$ rather than $c\,[u\rangle_W$.

LEMMA 4.5. *Let $W=(E, C)$ be an event structure and let $c \in C$ and $u_1, u_2 \in P_{\text{fin}}(E)$ be such that $c\,[u_1\rangle$, $c\,[u_2\rangle$ and $(c \cup u_1) \uparrow (c \cup u_2)$. Then $c\,[u_1 \cup u_2\rangle$.*

*Proof.* Clearly, $c \cap (u_1 \cup u_2) = \emptyset$. Let $v \subseteq u_1 \cup u_2$. From $c\,[u_1\rangle$ and $c\,[u_2\rangle$ we know that $c \cup (v \cap u_1) \in C$ and $c \cup (v \cap u_2) \in C$ and therefore $c \cup v = (c \cup (v \cap u_1)) \cup (c \cup (v \cap u_2)) \in C$ by (W2). ∎

We now define the underlying language of the generalized trace language we associate with an event structure. The steps in the step sequences of this language consist of sets of events which are enabled at the current configuration. Similar to the approach followed for Petri nets we also define a corresponding transition relation. This transition relation gives for each step sequence the configuration to which a step sequence leads, which is the set of elements occurring in the step sequence.

DEFINITION 4.6. Let $W=(E, C)$ be an event structure. The set $S_W \subseteq E^{\#}$ of *step sequences of* $W$ and the *step transition relation* $\Rightarrow_W \subseteq \{\emptyset\} \times S_W \times C$ are the least sets such that:

(i) $\underline{0} \in S_W$ and $(\emptyset, \underline{0}, \emptyset) \in \Rightarrow_W$.

(ii) if $\rho \in S_W$ and $(\emptyset, \rho, c) \in \Rightarrow_W$ and if $u \in P_{\text{fin}}(E)$ is such that $c\,[u\rangle_W$, then $\rho u \in S_W$ and $(\emptyset, \rho u, c \cup u) \in \Rightarrow_W$.

We write $\emptyset \overset{\rho}{\Longrightarrow}_W c$ rather than $(\emptyset, \rho, c) \in \Rightarrow_W$ in the sequel. Furthermore we may omit the index $W$ in $S_W$ and $\Rightarrow_W$ if $W$ is clear from the context.

Note that by (W1), for every configuration $c$ there exists $\rho \in S$ such that $\emptyset \overset{\rho}{\Longrightarrow} c$. Also note that given a configuration $c$, all $\rho \in S$ such that $\emptyset \overset{\rho}{\Longrightarrow} c$ satisfy $\text{alph}(\rho) = c$. Note also that in a step sequence of an event structure each event occurs at most once by the definition of the enabling relation.

LEMMA 4.7. *Let $W=(E, C)$ be an event structure and let $\rho \in S$ and $c \in C$ be such that $\text{alph}(\rho) \subseteq c$. Then there exists $\sigma \in E^{\#}$ such that $\emptyset \overset{\rho\sigma}{\Longrightarrow} c$.*

*Proof.* Let $c' = \text{alph}(\rho)$ and let $k = \#(c - c')$. We proceed by induction on $k$.

If $k=0$ or $k=1$ the result follows at once. Hence assume that $k>1$. Let $e_1, e_2 \in c - c'$ be such that $e_1 \neq e_2$. Then by Lemma 4.2 there exists $c'' \in C$ with $c'' \subseteq c$ such that $e_1 \in c''$ iff $e_2 \notin c''$. Assume without loss of generality that $e_1 \in c''$ so that $e_2 \notin c''$. Now $c' \subseteq c$ and $c'' \subseteq c$. Hence, by (W2), $c' \cup c'' \in C$. Moreover $c' \cup c'' \subseteq c$. We even have $c' \cup c'' \subset c$, because $e_2 \in c - (c' \cup c'')$. Furthermore note that $c' \subset c' \cup c''$, because $e_1 \in c'' - c'$. As a result, $\#((c' \cup c'') - c') \leqslant k$ and $\#(c - (c' \cup c'')) < k$. Hence by the induction hypothesis, there exist $\sigma_1, \sigma_2 \in E$ such that $\emptyset \overset{\rho\sigma_1}{\Longrightarrow} c' \cup c''$ and $\emptyset \overset{\rho\sigma_1\sigma_2}{\Longrightarrow} c$. The required result now follows with $\sigma = \sigma_1\sigma_2$. ∎

Analogous to the approach followed for Petri nets we now associate a concurrency alphabet $(E, \mathbf{I}_W)$ with every event structure $W=(E, C)$. The independent multisets are again the steps in step sequences. Thus these independent multisets are sets.

DEFINITION 4.8. Let $W=(E, C)$ be an event structure. The *independence relation of* $W$ is the set $\mathbf{I}_W \subseteq E^{\#} \times E^{\otimes}$ with $\mathbf{I}_W = \{(\rho, u) \mid \rho u \in S \text{ with } \rho \in E^{\#} \text{ and } u \in E^{\otimes}\}$.

Since $(\rho, u) \in \mathbf{I}_W$ iff $\text{alph}(\rho)[u\rangle$, the context in the independence relation is here determined by the configuration to which a step sequence leads, rather than by the step sequence itself.

It is easy to verify that the independence relation of an event structure satisfies the properties (D1), (D2), and (D3) mentioned in Section 1.

In what follows, we write $\doteq_W$, $\equiv_W$, $\leqslant_W$, and $[\rho]_W$ instead of $\doteq_{\mathbf{I}_W}$, $\equiv_{\mathbf{I}_W}$, $\leqslant_{\mathbf{I}_W}$, and $[\rho]_{\mathbf{I}_W}$, respectively.

LEMMA 4.9. *Let $W=(E, C)$ be an event structure. Then $S$ is consistent with respect to $(E, \mathbf{I}_W)$.*

*Proof.* This follows easily from the observation that $\mathbf{I}_W$ satisfies the property (D2) mentioned in Section 1. (See also the proof of Lemma 2.8.) ∎

For trace languages in general, whenever step sequences $\rho$ and $\rho'$ are equivalent, then $\text{num}_a(\rho) = \text{num}_a(\rho')$ for every element $a$ from the underlying alphabet. For step sequences generated by an event structure we also have the reverse as the following lemma shows. As a consequence all step sequences leading to a given configuration form a trace.

LEMMA 4.10. *Let $W=(E, C)$ be an event structure and let $\rho_1, \rho_2 \in S$ be such that $\text{alph}(\rho_1) = \text{alph}(\rho_2)$. Then $\rho_1 \equiv_W \rho_2$.*

*Proof.* The proof uses induction on $k = \#\text{alph}(\rho_1) - \#\text{alph}(\rho)$, where $\rho \in S$ is the maximal common prefix of $\rho_1$ and $\rho_2$.

If $k=0$ then $\rho = \rho_1 = \rho_2$, so trivially $\rho_1 \equiv_W \rho_2$. Now assume that $k>0$. Let $u_1, u_2 \in P_{\text{fin}}(E)$ and $\rho_1', \rho_2' \in E^{\#}$ be such that $u_1, u_2 \neq \emptyset$, $\rho_1 = \rho u_1 \rho_1'$ and $\rho_2 = \rho u_2 \rho_2'$. Then $\text{alph}(\rho)[u_1\rangle$ and $\text{alph}(\rho)[u_2\rangle$, so by Lemma 4.5 we know that $\text{alph}(\rho)[u_1 \cup u_2\rangle$. Hence $(\rho, u_1 \cup u_2) \in \mathbf{I}_W$ and $c = \text{alph}(\rho) \cup (u_1 \cup u_2) \in C$. Clearly, $c \subseteq \text{alph}(\rho_1)$, so by Lemma 4.7 there exists $\sigma \in E^{\#}$ such that $\emptyset \overset{\rho(u_1 \cup u_2)\sigma}{\Longrightarrow} \text{alph}(\rho_1)$. Hence $\rho(u_1 \cup u_2)\sigma \in S$. Furthermore we have $\rho(u_1 \cup u_2)\sigma \doteq_W \rho u_1((u_1 \cup u_2) - u_1)\sigma \doteq_W \rho u_2((u_1 \cup u_2) - u_2)\sigma$. So by the consistency of $S$, $\rho u_1((u_1 \cup u_2) - u_1)\sigma$,

$\rho u_2((u_1 \cup u_2) - u_2)\,\sigma \in S$ as well. We can now apply the induction hypothesis to $\rho u_1 \rho_1'$ and $\rho u_1((u_1 \cup u_2) - u_1)\,\sigma$ and to $\rho u_2 \rho_2'$ and $\rho u_2((u_1 \cup u_2) - u_2)\sigma$, because $\#\text{alph}(\rho u_1) > \#\text{alph}(\rho)$ and $\#\text{alph}(\rho u_2) > \#\text{alph}(\rho)$. Hence $\rho_1 = \rho u_1 \rho_1' \equiv_W \rho u_1((u_1 \cup u_2) - u_1)\,\sigma \doteq_W \rho u_2((u_1 \cup u_2) - u_2)\,\sigma \equiv_W \rho u_2 \rho_2' = \rho_2$. ∎

We are now ready to give a trace semantics for event structures.

DEFINITION 4.11. Let $W = (E, C)$ be an event structure.

(i) $\mathbf{TR}_W = (S, E, \mathbf{I}_W)$ is the *trace language generated by W*.

(ii) The poset $(\mathbf{TR}_W, \leqslant_W)$ is the *trace behaviour of W*.

Note that part (i) of the above definition is well defined because of Lemma 4.9.

From the observations made earlier we know that there is a 1–1 correspondence between the configurations of an event structure $W = (E, C)$ and the traces from $\mathbf{TR}_W$. Define trace($c$) as $\{\rho \in S \mid \varnothing \stackrel{\rho}{\Longrightarrow} c\}$ for each $c \in C$. Thus trace($c$) $\in \mathbf{TR}_W$ and $\bigcup \text{alph}(\text{trace}(c)) = c$. Conversely, for every $\rho \in S$, alph($\rho$) $\in C$ and trace(alph($\rho$)) = [$\rho$]. It is easy to see now that the set inclusion $\subseteq$ in $C$ and the prefix ordering $\leqslant_W$ in $\mathbf{TR}_W$ also correspond: $c \subseteq c'$ iff trace($c$) $\leqslant_W$ trace($c'$) (cf. Lemma 4.7).

EXAMPLE 4.3 (Continued). The trace behaviour of the given event structure is depicted in Fig. 8.

We now want to give a characterization of the trace languages associated with event structures. The notion of an *ES-trace language* is defined similarly to the notion of a PN-trace language.

DEFINITION 4.12. A trace language Tr is an ES-trace language if there exists an event structure $W$ such that $\text{Tr} \approx \mathbf{TR}_W$.

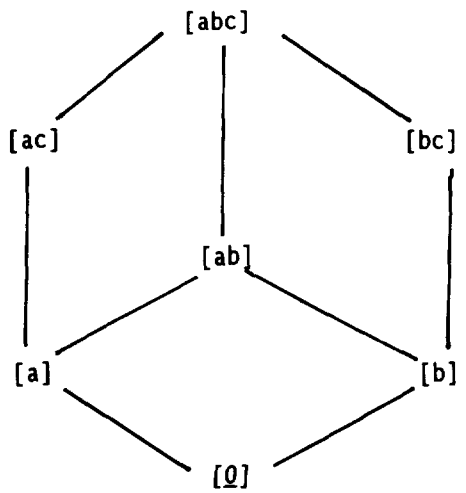


FIGURE 8

We prove that ES-trace languages $(L, \Sigma, \mathbf{I})$ are characterized by the axioms (PN1), (PN2), and (PN3) together with three new axioms (ES1), (ES2), and (ES3).

(ES1) $\rho \in L \Rightarrow$ for all $a \in \Sigma$: $\text{num}_a(\rho) \leqslant 1$ (Unique occurrence property).

(ES2) $\rho, \rho' \in L \Rightarrow (\text{alph}(\rho) \subseteq \text{alph}(\rho') \Rightarrow [\rho] \leqslant [\rho'])$ (Prefix-consistency).

(ES3) $\rho u_1 \rho_1,\ \rho u_2 \rho_2 \in L \Rightarrow (\rho u_1 \rho_1 \equiv \rho u_2 \rho_2 \Rightarrow \rho(\text{alph}(u_1 + u_2)) \in L)$ (Forward diamond property).

Note that in trace languages satisfying the unique occurrence property, the multisets which constitute the steps may also be viewed as sets. Hence in the presence of (ES1) we may also write $u_1 \cup u_2$ instead of $\text{alph}(u_1 + u_2)$ in (ES3).

Using Theorem 1.6 it is easy to see that (ES1), (ES2), and (ES3) are preserved under isomorphisms.

LEMMA 4.13. *Let* $(L_i, \Sigma_i, \mathbf{I}_i)$, $i = 1, 2$, *be a pair of trace languages such that* $(L_1, \Sigma_1, \mathbf{I}_1) \approx (L_2, \Sigma_2, \mathbf{I}_2)$. *Then, for each* $i = 1, 2, 3$, $(L_1, \Sigma_1, \mathbf{I}_1)$ *satisfies* (ES$i$) *iff* $(L_2, \Sigma_2, \mathbf{I}_2)$ *satisfies* (ES$i$).

Using this lemma we can prove one half of the result we are after.

LEMMA 4.14. *Let* $(L, \Sigma, \mathbf{I})$ *be an ES-trace language. Then* $(L, \Sigma, \mathbf{I})$ *satisfies the axioms* (PN1), (PN2), (PN3), (ES1), (ES2), *and* (ES3).

*Proof.* By Lemma 4.13 and Lemma 3.4 it is sufficient to prove that for every event structure $W = (E, C)$, the trace language $(S, E, \mathbf{I}_W)$ satisfies these six axioms.

It is easy to see that $(S, E, \mathbf{I}_W)$ satisfies the axioms (PN1), (PN2), (PN3), and (ES1). Furthermore (ES3) is satisfied because of Lemma 4.5.

In order to prove (ES2), let $\rho, \rho' \in S$ be such that $\text{alph}(\rho) \subseteq \text{alph}(\rho')$. Then by Lemma 4.7 there exists $\sigma \in E^*$ such that $\rho\sigma \in S_W$ and $\text{alph}(\rho\sigma) = \text{alph}(\rho')$. We can now conclude that $\rho\sigma \equiv_W \rho'$ by Lemma 4.10. ∎

In order to prove the converse result we proceed as follows.

For a trace language $\text{Tr} = (L, \Sigma, \mathbf{I})$ which satisfies the axioms (PN1), (PN2), (PN3), (ES1), (ES2), and (ES3) we define the structure $W_{\text{Tr}} = (E_{\text{Tr}}, C_{\text{Tr}})$, where $E_{\text{Tr}} = \Sigma$ and $C_{\text{Tr}} = \{\text{alph}(\rho) \mid \rho \in L\}$.

We prove that Tr is an ES-trace language by showing that $W_{\text{Tr}}$ is an event structure with $\text{Tr} \approx \mathbf{TR}_{W_{\text{Tr}}}$.

LEMMA 4.15. *Let* $\text{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1), (PN2), (PN3), (ES1), (ES2), *and* (ES3). *Then* $W_{\text{Tr}}$ *is an event structure.*

*Proof.* First we prove requirement (W1) from Definition 4.1.

Suppose $c \in C_{\text{Tr}}$ is such that $c \neq \varnothing$. Let $\rho \in L$ be such that $\text{alph}(\rho) = c$. Since $\text{alph}(\rho) \neq \varnothing$ we can write $\rho$ as $\rho = \rho' u$

with $\varnothing \neq u \in \Sigma^{\otimes}$. Since Tr satisfies the unique occurrence property, we may also view $u$ as a set. Let $a \in u$. By the substep property (PN3), $(\rho', u) \in \mathbf{I}$ and therefore we have $\rho'(u - \{a\})\{a\} \doteq_{\mathbf{I}} \rho' u$. From the consistency of $L$ we now know that $\rho'(u - \{a\})\{a\} \in L$ and by prefix-closedness of $L$ we then get $\rho'(u - \{a\}) \in L$. By the unique occurrence property, we know that $a \notin \mathrm{alph}(\rho')$. Thus $\mathrm{alph}(\rho'(u - \{a\})) = \mathrm{alph}(\rho) - \{a\} \in C_{\mathrm{Tr}}$. This proves that $(E_{\mathrm{Tr}}, C_{\mathrm{Tr}})$ satisfies the requirement (W1) from Definition 4.1.

In order to prove (W2), suppose that $c, c' \in C_{\mathrm{Tr}}$ are such that $c \uparrow c'$ with $c'' \in C_{\mathrm{Tr}}$ such that $c \subseteq c''$ and $c' \subseteq c''$. We must prove that $c \cup c' \in C_{\mathrm{Tr}}$.

Let $\rho, \rho', \rho'' \in L$ be such that $\mathrm{alph}(\rho) = c$, $\mathrm{alph}(\rho') = c'$, and $\mathrm{alph}(\rho'') = c''$. According to the definition of $C_{\mathrm{Tr}}$ such $\rho$, $\rho'$, and $\rho''$ exist. We show that there exists $\tilde{\rho} \in L$ such that $\mathrm{alph}(\tilde{\rho}) = \mathrm{alph}(\rho) \cup \mathrm{alph}(\rho')$. Let $\rho_0$ be the longest common prefix of $\rho$ and $\rho'$. So $\rho = \rho_0\sigma$ and $\rho' = \rho_0\sigma'$ for some $\sigma, \sigma' \in \Sigma^{\#}$. By the unique occurrence property of $L$ we know that $\mathrm{num}_a(\sigma) \cdot \mathrm{num}_a(\sigma') \leqslant 1$, for all $a \in \Sigma$, and $\mathrm{alph}(\rho_0) \cap \mathrm{alph}(\sigma) = \mathrm{alph}(\rho_0) \cap \mathrm{alph}(\sigma') = \varnothing$.

Let $k = \#\mathrm{alph}(\sigma) + \#\mathrm{alph}(\sigma')$. We proceed by induction on $k$. If $k = 0$, then $\rho = \rho'$ and so $\tilde{\rho} = \rho$ is as required. Now assume that $k > 0$. If $\#\mathrm{alph}(\sigma) = 0$ or $\#\mathrm{alph}(\sigma') = 0$, then we are done, because in that case $c \subseteq c'$ or $c' \subseteq c$, respectively. So assume that $\sigma = u\sigma_1$ and $\sigma' = u'\sigma_1'$ for some $u, u' \in \Sigma^{\otimes}$ and $\sigma_1, \sigma_1' \in \Sigma^{\#}$ with $u \neq \underline{0} \neq u'$ and $u \neq u'$. Since $\mathrm{alph}(\rho) \subseteq \mathrm{alph}(\rho'')$ and $\mathrm{alph}(\rho') \subseteq \mathrm{alph}(\rho'')$, we have by (ES2) that $\rho\sigma_2 \equiv_{\mathbf{I}} \rho''$ and $\rho'\sigma_2' \equiv_{\mathbf{I}} \rho''$ for certain $\sigma_2, \sigma_2' \in \Sigma^{\#}$. Hence $\rho_0 u\sigma_1\sigma_2 \equiv_{\mathbf{I}} \rho_0 u'\sigma_1'\sigma_2'$. From (ES1) and (ES3) it now follows that $\rho_0(u \cup u') \in L$. Now we have $(\rho_0, u \cup u') \in \mathbf{I}$ by (PN3). Hence $\rho_0 u(u' - u) \doteq_{\mathbf{I}} \rho_0(u \cup u')$ and thus $\rho_0 u(u' - u) \in L$ by the consistency of $L$. Now let us first compare $\rho = \rho_0 u\sigma_1$ and $\rho_0 u(u' - u)$. Clearly, $\#\mathrm{alph}(\sigma_1) + \#(u' - u) < \#\mathrm{alph}(\sigma) + \#\mathrm{alph}(\sigma')$ and so we can use our induction hypothesis to conclude that there is a $\hat{\rho} \in L$ such that $\mathrm{alph}(\hat{\rho}) = \mathrm{alph}(\rho_0 u\sigma_1) \cup \mathrm{alph}(\rho_0 u(u' - u)) = \mathrm{alph}(\rho) \cup (u' - u)$. By (ES2) we have in addition, that $\hat{\rho} \equiv_{\mathbf{I}} \rho_0 u'\tau$, where $\tau \in \Sigma^{\#}$ is such that $\mathrm{alph}(\tau) = \mathrm{alph}(u\sigma_1) - u'$. By the consistency of $L$, $\rho_0 u'\tau \in L$. Next we compare $\rho_0 u'\tau$ and $\rho_0 u'\sigma_1'$. Now $\#\mathrm{alph}(\tau) + \#\mathrm{alph}(\sigma_1') < \#\mathrm{alph}(\sigma) + \#\mathrm{alph}(\sigma') = k$. By the induction hypothesis there exists a $\tilde{\rho} \in L$ such that $\mathrm{alph}(\tilde{\rho}) = \mathrm{alph}(\rho_0 u'\tau) \cup \mathrm{alph}(\rho_0 u'\sigma_1') = \mathrm{alph}(\rho_0) \cup \mathrm{alph}(\sigma_1) \cup \mathrm{alph}(\sigma_1') \cup u \cup u' = \mathrm{alph}(\rho) \cup \mathrm{alph}(\rho')$. ∎

Given a trace language $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ satisfying (PN1), (PN2), (PN3), (ES1), (ES2), and (ES3), we want to prove that $\mathrm{Tr} \approx \mathbf{TR}_{W_{\mathrm{Tr}}}$ where the required isomorphism is the identity function.

In order to do this, we first show that the underlying languages of both trace languages are the same.

LEMMA 4.16. *Let* $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1), (PN2), (PN3), (ES1), (ES2), *and* (ES3). *Then* $L = S_{W_{\mathrm{Tr}}}$.

*Proof.* Let $\rho \in L$. By the definition of $W_{\mathrm{Tr}}$ we then have that $\mathrm{alph}(\rho) \in C_{\mathrm{Tr}}$.

If $\rho = \underline{0}$, then we have $\rho \in S_{W_{\mathrm{Tr}}}$, because $W_{\mathrm{Tr}}$ is an event structure. Now assume that $\rho = \rho' u$ for some $\rho' \in L$ and $\underline{0} \neq u \in \Sigma^{\otimes}$. Then $\mathrm{alph}(\rho') \in C_{\mathrm{Tr}}$ and the unique occurrence property guarantees that $u$ is a set such that $\mathrm{alph}(\rho') \cap u = \varnothing$. Let $v \subseteq u$. By the substep property we have $(\rho', u) \in \mathbf{I}$ and therefore $\rho' u \doteq_{\mathbf{I}} \rho' v(u - v)$. Hence $\rho' v(u - v) \in L$ because $L$ is consistent. From prefix-closedness we now know that $\rho' v \in L$ and therefore $\mathrm{alph}(\rho') \cup v = \mathrm{alph}(\rho' v) \in C_{\mathrm{Tr}}$. Since $v$ is an arbitrary subset of $u$ this proves that $\mathrm{alph}(\rho')[u\rangle_{W_{\mathrm{Tr}}}$. Thus $\rho = \rho' u \in S_{W_{\mathrm{Tr}}}$.

Now in order to prove that $S_{W_{\mathrm{Tr}}} \subseteq L$, let $\rho \in S_{W_{\mathrm{Tr}}}$. By induction on $k = \#\mathrm{alph}(\rho)$ we prove that $\rho \in L$.

If $k = 0$, then $\rho = \underline{0}$, so $\rho \in L$ by the non-emptiness and prefix-closedness of $L$. Now assume that $k > 0$. First note that by the definition of $C_{\mathrm{Tr}}$, there exists $\rho_1 \in L$ such that $\mathrm{alph}(\rho) = \mathrm{alph}(\rho_1)$. Let $\rho' \in S_{W_{\mathrm{Tr}}}$ and $\underline{0} \neq u \in (E_{\mathrm{Tr}})^{\otimes}$ be such that $\rho = \rho' u$. Then $\mathrm{alph}(\rho')[u\rangle_{W_{\mathrm{Tr}}}$. Since $\#\mathrm{alph}(\rho') < k$ we have $\rho' \in L$ by the induction hypothesis. First assume that $\#u = 1$. Then because $\mathrm{alph}(\rho') \subseteq \mathrm{alph}(\rho_1)$, there exists $\sigma \in \Sigma^{\#}$ such that $\rho'\sigma \equiv_{\mathbf{I}} \rho_1$ by (ES2). Clearly, $\sigma = u$ must hold, so $\rho' u \in L$, because $L$ is consistent. Now assume that $\#u > 1$. Let $a \in u$. Since $\rho'\{a\} \in S_{W_{\mathrm{Tr}}}$ and $\rho'(u - \{a\}) \in S_{W_{\mathrm{Tr}}}$, we know that $\rho'\{a\} \in L$ and $\rho'(u - \{a\}) \in L$ by the induction hypothesis. Moreover $\mathrm{alph}(\rho'\{a\}) \subseteq \mathrm{alph}(\rho_1)$ and $\mathrm{alph}(\rho'(u - \{a\})) \subseteq \mathrm{alph}(\rho_1)$. Then by (ES2) there exist $\sigma_1, \sigma_2 \in \Sigma^{\#}$ such that $\rho'\{a\}\,\sigma_1 \equiv_{\mathbf{I}} \rho_1$ and $\rho'(u - \{a\})\,\sigma_2 \equiv_{\mathbf{I}} \rho_1$. Hence $\rho'((u - \{a\}) \cup \{a\}) = \rho' u \in L$ by (ES3). ∎

In order to prove that a trace language $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ satisfying (PN1), (PN2), (PN3), (ES1), (ES2), and (ES3), and $\mathbf{TR}_{W_{\mathrm{Tr}}}$ are isomorphic, the only thing left to prove is that $\mathbf{I}$ and $\mathbf{I}_{W_{\mathrm{Tr}}}$ are the same.

LEMMA 4.17. *Let* $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1), (PN2), (PN3), (ES1), (ES2), *and* (ES3). *Then* $\mathrm{Tr} \approx \mathbf{TR}_{W_{\mathrm{Tr}}}$.

*Proof.* In order to prove that the identity function on $\Sigma$ is an isomorphism from Tr to $\mathbf{TR}_{W_{\mathrm{Tr}}}$ it is sufficient—according to Theorem 1.6 and Lemma 4.16—to prove that $\mathbf{I} = \mathbf{I}_{W_{\mathrm{Tr}}}$. This, however, follows immediately from Lemma 4.16, the substep property (PN3), and the definition of $\mathbf{I}_{W_{\mathrm{Tr}}}$. ∎

From Lemma 4.13 and Lemma 4.17 we get the following result.

THEOREM 4.18. *A trace language* Tr *is an ES-trace language iff* Tr *satisfies the axioms* (PN1), (PN2), (PN3), (ES1), (ES2), *and* (ES3).

Starting from an event structure $W$, we may associate the trace language $\mathbf{TR}_W$ with $W$ and then apply the construction above and obtain an event structure $\hat{W}$. It is easy to see

that $W = \hat{W}$. Hence the above described correspondence between trace languages satisfying the six axioms and event structures is a 1–1 correspondence.

Next we prove that the class of ES-trace languages is included in the class of PN-trace languages. The proof is based on the existence of certain regions, which guarantee that ES-trace languages satisfy the forward closure property (PN4).

THEOREM 4.19. *Every ES-trace language is a PN-trace language.*

*Proof.* Let $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ be an ES-trace language. By Theorem 4.18 we know that Tr satisfies the axioms (PN1), (PN2), (PN3), (ES1), (ES2), and (ES3). By Theorem 3.12 and Theorem 4.18 it is now sufficient to prove that Tr satisfies the axiom (PN4).

Let $\rho \in L$ and $u \in \Sigma^{\otimes}$ be such that $\rho u \notin L$. Hence $u \neq \underline{0}$. It must be proved that there exists $r \in \mathbf{R}_{\mathrm{Tr}}$ such that $r(\rho) < \sum_{b \in \Sigma} u(b) \cdot {}^{r}b$. Two cases can be distinguished.

First assume that $\mathrm{num}_a(\rho u) > 1$ for some $a \in \Sigma$. Then define $r\langle a\rangle : L \cup \Sigma \to \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$ as follows:

(i) For all $b \in \Sigma$: $r\langle a\rangle(b) = \begin{cases} (1,0) & \text{if } b = a \\ (0,0) & \text{otherwise.} \end{cases}$

(ii) For all $\rho' \in L$: $r\langle a\rangle(\rho') = 1 - \#(\mathrm{alph}(\rho') \cap a)$.

CLAIM 1. $r\langle a\rangle \in \mathbf{R}_{\mathrm{Tr}}$.

Assume for the moment that Claim 1 holds. Now $\mathrm{num}_a(\rho) \leqslant 1$ by the unique occurrence property. If $\mathrm{num}_a(\rho) = 0$, then $u(a) \geqslant 2$, and so $r\langle a\rangle(\rho) = 1 < 2 \leqslant \sum_{b \in \Sigma} u(b) \cdot {}^{r\langle a\rangle}b$. If $\mathrm{num}_a(\rho) = 1$ then $u(a) \geqslant 1$, and so $r\langle a\rangle(\rho) = 0 < 1 \leqslant \sum_{b \in \Sigma} u(b) \cdot {}^{r\langle a\rangle}b$. Hence by Claim 1 $r\langle a\rangle$ is as required.

Now assume that $\mathrm{num}_a(\rho u) \leqslant 1$ for all $a \in \Sigma$. Hence $u$ may be viewed as a set. Then define $r\langle u, \rho\rangle : L \cup \Sigma \to \mathbb{N} \cup (\mathbb{N} \times \mathbb{N})$ as follows:

(i) For all $b \in \Sigma$: $r\langle u, \rho\rangle(b) = \begin{cases} (1,0) & \text{if } b \in \mathrm{alph}(\rho) \\ (1,1) & \text{if } b \in u \\ (0,1) & \text{otherwise.} \end{cases}$

(ii) For all $\sigma \in L$: $r\langle u, \rho\rangle(\sigma) = \#\mathrm{alph}(\rho) + \#u - 1 + \sum_{b \in \mathrm{alph}(\sigma)} (b^{r\langle u,\rho\rangle} - {}^{r\langle u,\rho\rangle}b)$.

CLAIM 2. $r\langle u, \rho\rangle \in \mathbf{R}_{\mathrm{Tr}}$.

Then by Claim 2 $r = r\langle u, \rho\rangle$ is as required, because $r(\rho) = \#\mathrm{alph}(\rho) + \#u - 1 - \#\mathrm{alph}(\rho) = \#u - 1 < \#u = \sum_{b \in \Sigma} u(b) \cdot {}^{r}b$.

Hence if Claim 1 and Claim 2 hold, then Tr satisfies (PN4), and we are done.

*Proof of Claim* 1. To simplify the notation we write $r$ instead of $r\langle a\rangle$ in this proof. Clearly it suffices to prove that $r$ is a region of Tr.

Let $\rho', \rho'' \in L$ and $v \in \Sigma^{\otimes}$ be such that $\rho' \xrightarrow{v} \rho''$. By the consistency of $L$ and the fact that $\rho'' \in L$ we know that $\rho' v \in L$. By (ES1) we may view $v$ as a set. Now we have that $\mathrm{alph}(\rho') \cup v = \mathrm{alph}(\rho'')$ and $\mathrm{alph}(\rho') \cap v = \varnothing$. Therefore $r(\rho') = 1 - \#(\mathrm{alph}(\rho') \cap a) = r(\rho'') + \#(a \cap v) \geqslant \#(a \cap v) = \sum_{b \in v} {}^{r}b$ and $r(\rho'') = r(\rho') - \#(a \cap v) = r(\rho') + \sum_{b \in v} (b^{r} - {}^{r}b)$. Hence $r$ is a region of Tr. This finishes the proof of Claim 1.

*Proof of Claim* 2. In order to simplify the notation, we write $r$ instead of $r\langle u, \rho\rangle$ in this proof. Since $u \neq \varnothing$, it suffices to prove that $r$ is a region of Tr.

Suppose $\sigma, \sigma' \in L$ and $v \in \Sigma^{\otimes}$ are such that $\sigma \xrightarrow{v} \sigma'$. By the definition of $\to$, we have $\sigma v \equiv \sigma'$. Hence by the consistency of $L$, $\sigma v \in L$ and $\mathrm{num}_a(\sigma v) \leqslant 1$ for all $a \in \Sigma$ by the unique occurrence property; since $\mathrm{alph}(\sigma') = \mathrm{alph}(\sigma) \cup \mathrm{alph}(v)$, it now follows from the definition of $r$ that $r(\sigma') = r(\sigma) + \sum_{a \in v} (a^{r} - {}^{r}a)$. Thus in order to prove that $r$ is a region we merely need to prove that $r(\sigma) \geqslant \sum_{a \in v} {}^{r}a$.

Let $n = \#(v \cap (\mathrm{alph}(\rho) \cup u)) = \sum_{a \in v} {}^{r}a$. Then we must prove that $r(\sigma) \geqslant n$. Now set $k = \#(\mathrm{alph}(\sigma) \cap u)$, $j = \#(\mathrm{alph}(\sigma) \cap \mathrm{alph}(\rho))$, and $m = \#(\mathrm{alph}(\sigma) \cap (\Sigma - (\mathrm{alph}(\rho) \cup u)))$. Then from the unique occurrence property it follows that $n \leqslant \#\mathrm{alph}(\rho) + \#u - k - j$. Moreover, by the definition of $r$, it is clear that $r(\sigma) = \#\mathrm{alph}(\rho) + \#u - 1 + k + m - k - j = \#\mathrm{alph}(\rho) + \#u - 1 + m - j$. Hence if $m + k \geqslant 1$ we are done. Therefore we assume in the rest of the proof that $m = k = 0$. In other words, we assume that $\mathrm{alph}(\sigma) \subseteq \mathrm{alph}(\rho)$. This leads to the equation $r(\sigma) = \#\mathrm{alph}(\rho) + \#u - 1 - \#\mathrm{alph}(\sigma)$. On the other hand, $n \leqslant \#\mathrm{alph}(\rho) + \#u - \#\mathrm{alph}(\sigma)$. If $n < \#\mathrm{alph}(\rho) + \#u - \#\mathrm{alph}(\sigma)$ then at once we get $r(\sigma) \geqslant n$. We now wish to argue that $n = \#\mathrm{alph}(\rho) + \#u - \#\mathrm{alph}(\sigma)$ leads to a contradiction.

To see this, suppose that $n = \#\mathrm{alph}(\rho) + \#u - \#\mathrm{alph}(\sigma)$. Now recall that $n = \#(v \cap (\mathrm{alph}(\rho) \cup u))$ and that $\mathrm{alph}(\rho) \cap u = \varnothing$. So let $v_1 = v \cap \mathrm{alph}(\rho)$ and $v_2 = v \cap u$. Then from $\mathrm{alph}(\sigma) \cap v = \varnothing$ and $\mathrm{alph}(\sigma) \subseteq \mathrm{alph}(\rho)$ it follows that $v_1 = \mathrm{alph}(\rho) - \mathrm{alph}(\sigma)$ and $v_2 = u$. Let $v_3 = v - (v_1 \cup v_2)$.

Then $\sigma v \in L$ implies that $(\sigma, v) \in \mathbf{I}$ by the substep property and so $\sigma v \doteq \sigma(v_1 \cup u)\, v_3$. Now we must also have $\sigma(v_1 \cup u)\, v_3 \in L$ by the consistency of $L$. The prefix-closedness of $L$ guarantees that $\sigma(v_1 \cup u) \in L$ as well. Again by the substep property, $(\sigma, v_1 \cup u) \in \mathbf{I}$ and therefore $\sigma(v_1 \cup u) \doteq \sigma v_1 u$ because $v_1 \cap u = \varnothing$. By the consistency of $L$ we then have $\sigma v_1 u \in L$.

From the prefix-closedness of $L$ we now know that $\sigma v_1 \in L$. Clearly, $\mathrm{alph}(\sigma v_1) = \mathrm{alph}(\rho)$, so by (ES2) we have $\sigma v_1 \equiv \rho$. Since $\rho u \notin L$ this leads to the contradiction that $\sigma v_1 u \notin L$. This proves that $n = \#\mathrm{alph}(\rho) + \#u - \#\mathrm{alph}(\sigma)$

is not possible, so $r(\sigma) \geq n$. This finishes the proof of Claim 2. ▮

Hence for every event structure there exists a Petri net which yields an isomorphic trace language. The converse of this theorem is not true. Let $N_1$, $N_6$, and $N_5$ be the Petri nets from Example 2.5 and Example 2.10. Then $\mathbf{TR}_{N_i}$, $i = 1, 6, 5$ are PN-trace languages, but $\mathbf{TR}_{N_1}$ does not satisfy (ES1), $\mathbf{TR}_{N_6}$ does not satisfy (ES2), and $\mathbf{TR}_{N_5}$ does not satisfy (ES3). Hence $\mathbf{TR}_{N_1}$, $\mathbf{TR}_{N_6}$, and $\mathbf{TR}_{N_5}$ are not ES-trace languages.

To conclude this section we want to characterize an interesting subclass of event structures called *stable* event structures [Winskel, 1987a] in terms of our trace languages. Stable event structures are important because they impose a partial order of causal dependencies on the events in each configuration; each event in a configuration is enabled by a *minimal* set of events. From Winskel's work [Winskel, 1987a] it follows that stable event structures "correspond" to the behaviours of 1-safe Petri nets.

DEFINITION 4.20. Let $W = (E, C)$ be an event structure. $W$ is stable if it satisfies

(W3) For all $c, c' \in C$, if $c \uparrow c'$ then $c \cap c' \in C$.

The event structure from Example 4.3 is not stable, because $\{a, c\} \uparrow \{b, c\}$, but $\{a, c\} \cap \{b, c\} = \{c\} \notin C$.

The notion of a stable ES-trace language is defined in the obvious way.

DEFINITION 4.21. A trace language Tr is a *stable ES-trace language* if there exists a stable event structure $W$ such that $\mathrm{Tr} \approx \mathbf{TR}_W$.

In the remaining part of this section we prove that stable ES-trace languages are characterized by the axioms (PN1), (PN2), (PN3), (ES1), (ES2), and (ES3) and the following axiom:

(ES4) $\rho_1 u_1,\ \rho_2 u_2 \in L \Rightarrow [\rho_1 u_1 \equiv \rho_2 u_2 \Rightarrow$ there exists $\rho \in L$ such that $\rho(\mathrm{alph}(u_1 + u_2)) \equiv \rho_1 u_1]$ (Backward diamond property).

In the presence of the unique occurrence property, $u_1$ and $u_2$ are once again sets, so then we can also write $u_1 \cup u_2$ instead of $\mathrm{alph}(u_1 + u_2)$.

Note that the trace language of the non-stable event structure of Example 4.3 does not satisfy (ES4). In order to prove that stable ES-trace languages satisfy this new axiom, we first observe that (ES4) is preserved under isomorphisms.

LEMMA 4.22. *Let* $\mathrm{Tr}_i$, $i = 1, 2$, *be a pair of trace languages such that* $\mathrm{Tr}_1 \approx \mathrm{Tr}_2$. *Then* $\mathrm{Tr}_1$ *satisfies* (ES4) *iff* $\mathrm{Tr}_2$ *satisfies* (ES4).

LEMMA 4.23. *Let* Tr *be a stable ES-trace language. Then* Tr *satisfies the axioms* (PN1), (PN2), (PN3), *and* (ES1) *through* (ES4).

*Proof.* By the previous lemma and Lemma 4.14 it is sufficient to prove that for every stable event structure $W$, the trace language $\mathbf{TR}_W$ satisfies (ES4).

Let $W = (E, C)$ be a stable event structure. Let $\rho_1, \rho_2 \in S$ and $u_1, u_2 \in P_{\mathrm{fin}}(E)$ with $\rho_1 u_1, \rho_2 u_2 \in S$, be such that $\rho_1 u_1 \equiv_W \rho_2 u_2$. Then $\mathrm{alph}(\rho_1), \mathrm{alph}(\rho_2) \in C$ and $\mathrm{alph}(\rho_1) \uparrow \mathrm{alph}(\rho_2)$. Hence by (W3) we know that $c = \mathrm{alph}(\rho_1) \cap \mathrm{alph}(\rho_2) \in C$. By the remark preceding Lemma 4.7 there exists $\rho \in S$ such that $\mathrm{alph}(\rho) = c$. We prove that $c[u_1 \cup u_2\rangle$. From $\mathrm{alph}(\rho_1)[u_1\rangle$ and $\mathrm{alph}(\rho_2)[u_2\rangle$ we have that $(u_1 \cup u_2) \cap c = \varnothing$. Let $v \subseteq u_1 \cup u_2$. We must prove that $c \cup v \in C$. From $\mathrm{alph}(\rho_1)[u_1\rangle$ and $\mathrm{alph}(\rho_2)[u_2\rangle$ we have that $\mathrm{alph}(\rho_1) \cup (u_1 \cap v) \in C$ and $\mathrm{alph}(\rho_2) \cup (u_2 \cap v) \in C$. Moreover $(\mathrm{alph}(\rho_1) \cup (u_1 \cap v)) \uparrow (\mathrm{alph}(\rho_2) \cup (u_2 \cap v))$ and so by (W3), $c \cup v = (\mathrm{alph}(\rho_1) \cup (u_1 \cap v)) \cap (\mathrm{alph}(\rho_2) \cup (u_2 \cap v)) \in C$. This proves that $c[u_1 \cup u_2\rangle$. Therefore we also have $\rho(u_1 \cup u_2) \in S$. Since $\mathrm{alph}(\rho(u_1 \cup u_2)) = \mathrm{alph}(\rho_1 u_1)$ we can conclude by Lemma 4.10 that $\rho_1 u_1 \equiv_W \rho(u_1 \cup u_2)$. ▮

In order to prove the converse result, we prove that for a trace language Tr satisfying the new set of axioms, the event structure $W_{\mathrm{Tr}}$ is stable.

LEMMA 4.24. *Let* $\mathrm{Tr} = (L, \Sigma, \mathbf{I})$ *be a trace language which satisfies the axioms* (PN1), (PN2), (PN3), *and* (ES1) *through* (ES4). *Then* $W_{\mathrm{Tr}} = (E_{\mathrm{Tr}}, C_{\mathrm{Tr}})$ *is a stable event structure.*

*Proof.* Let $c, c' \in C_{\mathrm{Tr}}$ be such that $c \uparrow c'$. Then $c'' = c \cup c' \in C_{\mathrm{Tr}}$ because of requirement (W2) in the definition of an event structure. We will prove that $c \cap c' \in C_{\mathrm{Tr}}$ by induction on $k = \#(c'' - c) + \#(c'' - c')$. Let $\rho, \rho', \rho'' \in L$ be such that $c = \mathrm{alph}(\rho)$, $c' = \mathrm{alph}(\rho')$, and $c'' = \mathrm{alph}(\rho'')$.

If $k = 0$ then $c = c'$, so trivially $c \cap c' \in C_{\mathrm{Tr}}$. Now assume that $k > 0$. If $c \subseteq c'$ or $c' \subseteq c$ then we are done, so assume that $c \not\subseteq c'$ and $c' \not\subseteq c$. Then by (ES1) and (ES2) there exist $u, u' \in P_{\mathrm{fin}}(\Sigma)$ and $\sigma, \sigma' \in \Sigma^*$ such that $u \neq \underline{0}$, $u' \neq \underline{0}$, $\rho\sigma u \equiv_{\mathbf{I}} \rho''$ and $\rho'\sigma' u' \equiv_{\mathbf{I}} \rho''$. By the consistency of $L$ we have $\rho\sigma u, \rho'\sigma' u' \in L$ and by prefix-closedness $\rho\sigma, \rho'\sigma' \in L$.

Now we will prove that $c'' - (u \cup u') \in C_{\mathrm{Tr}}$. Since $\rho\sigma u \equiv_{\mathbf{I}} \rho'\sigma' u'$ we get by (ES4) that there exists $\rho_1 \in L$ such that $\rho_1(u \cup u') \equiv_{\mathbf{I}} \rho\sigma u$. From the consistency of $L$ we know that $\rho_1(u \cup u') \in L$. By the unique occurrence property we now have that $\mathrm{alph}(\rho_1) = c'' - (u \cup u')$, so $c'' - (u \cup u') \in C_{\mathrm{Tr}}$. Then $\hat{c} = c \cup (c'' - (u \cup u')) \in C_{\mathrm{Tr}}$ by (W2). By the unique occurrence property $u \cap c = \varnothing$; hence $\hat{c} \subseteq c'' - u$. Since $\#((c'' - u) - c) < \#(c'' - c)$ and $\#((c'' - u) - (c'' - (u \cup u'))) \leq \#(c'' - c')$ by the fact that $u' \cap c' = \varnothing$, we have that $\#(\hat{c} - c) + \#(\hat{c} - (c'' - (u \cup u'))) < k$. Then by the induction hypothesis $c \cap (c'' - (u \cup u')) = c - u' \in C_{\mathrm{Tr}}$. Moreover $(c - u') \cup c' \subseteq c'' - u'$. Since $\#((c'' - u') - (c - u')) \leq \#(c'' - c)$ and $\#((c'' - u') - c') < \#(c'' - c')$ by

the fact that $u' \cap c' = \varnothing$, we have $\#((c'' - u') - (c - u')) + \#((c'' - u') - c') < k$. Hence by the induction hypothesis and the fact that $c' \cap u' = \varnothing$ we can conclude that $(c - u') \cap c' = c \cap c' \in C_{\mathrm{Tr}}$. ∎

From Lemma 4.23 and Lemma 4.24 we immediately obtain the following result.

THEOREM 4.25. *A trace language* Tr *is a stable ES-trace language iff* Tr *satisfies the axioms* (PN1), (PN2), (PN3), *and* (ES1) *through* (ES4).

## DISCUSSION

In this paper we have introduced a generalization of the classical traces model from Mazurkiewicz. In order to see that this model is embedded in our framework, consider generalized concurrency alphabets $(\Sigma, \mathbf{I})$ satisfying the following four conditions.

(T1) If $(\rho, u) \in \mathbf{I}$ then $u(a) \leqslant 1$ for all $a \in \Sigma$.

(T2) If $(\rho, u) \in \mathbf{I}$ then for all $\rho' \in \Sigma^{\#}$, $(\rho', u) \in \mathbf{I}$.

(T3) If $(\rho, u) \in \mathbf{I}$ then for all $v \leqslant u$, $(\rho, v) \in \mathbf{I}$.

(T4) If $u \in P_{\mathrm{fin}}(\Sigma)$ is such that for all $a, b \in u$ with $a \neq b$, $(\rho, \{a, b\}) \in \mathbf{I}$, then $(\rho, u) \in \mathbf{I}$.

Every (Mazurkiewicz) concurrency alphabet $(\Sigma, \mathbf{I})$, where $\mathbf{I} \subseteq \Sigma \times \Sigma$ is an irreflexive symmetric relation, may be viewed as a generalized concurrency alphabet $(\Sigma, \mathbf{I}_G)$ with $\mathbf{I}_G = \{(\rho, u) \mid \rho \in \Sigma^{\#},\ u \in P_{\mathrm{fin}}(\Sigma)$ such that for all $a, b \in u$ with $a \neq b$: $(a, b) \in \mathbf{I}\}$. This generalized concurrency alphabet satisfies the conditions (T1) through (T4).

Conversely, with every generalized concurrency alphabet $(\Sigma, \mathbf{I})$ satisfying (T1) through (T4), a Mazurkiewicz concurrency alphabet $(\Sigma, \mathbf{I}_M)$ can be associated with $\mathbf{I}_M = \{(a, b) \mid a, b \in \Sigma$ with $a \neq b$ and $(0, \{a, b\}) \in \mathbf{I}\}$. In this way we obtain a bijection between Mazurkiewicz concurrency alphabets and generalized concurrency alphabets satisfying (T1) through (T4).

With a Mazurkiewicz trace language $(L, \Sigma, \mathbf{I})$ we can now associate a generalized trace language $(L_G, \Sigma, \mathbf{I}_G)$, where $\mathbf{I}_G$ is as defined above and $L_G = \bigcup \{[\mathrm{mset}(x)]_{\mathbf{I}_G} \mid x \in L\}$, where mset: $\Sigma^* \to \Sigma^{\#}$ is the homomorphism defined by

$$(\mathrm{mset}(a))(b) = \begin{cases} 1 & \text{if } b = a \\ 0 & \text{if } b \neq a \end{cases} \quad \text{and} \quad \mathrm{mset}(\Lambda) = \underline{0}.$$

It is easy to see that $(L_G, \Sigma, \mathbf{I}_G)$ satisfies the following condition:

(T5) For all $\rho_1, \rho_2 \in \Sigma^{\#}$ and $u \in \Sigma^{\otimes}$, $\rho_1 u \rho_2 \in L_G \Rightarrow (\underline{0}, u) \in \mathbf{I}_G$.

Conversely, we can associate with a generalized trace language $(L, \Sigma, \mathbf{I})$ satisfying the conditions (T1) through (T5) the Mazurkiewicz trace language $(L_M, \Sigma, \mathbf{I}_M)$ where $\mathbf{I}_M$ is as defined above and $L_M = \{x \in \Sigma^* \mid \mathrm{mset}(x) \in L\}$. Thus we obtain a bijection between Mazurkiewicz trace languages and generalized trace languages satisfying (T1) through (T5). Apart from these obvious observations however, we do not have at present more to say on this topic.

The generalization of the classical trace model introduced in this paper provides a tool for the description of the behaviour of Petri nets. As remarked earlier only a particular kind of generalized concurrency alphabets have been used to achieve this aim. What is missing at this stage is a sound theoretical basis for the general theory of traces we propose.

In the last few years there have been other proposals for extending the theory of traces. Several of these, though differently motivated, focus on similar generalizations, in particular context-dependency or the use of steps (see, e.g., [Mazurkiewicz, 1989; Arnold, 1991; Rozoy, 1990; Vogler, 1991]). None of these related approaches, however, is geared towards obtaining the trace behaviour of a specific model of concurrency as is the case here. Hence useful connections are hard to establish between our approach and the cited related approaches.

Turning to two pieces of related work, both [Degano *et al.*, 1989] and [Best and Devillers, 1987] make valuable statements about the non-sequential behaviour of general Petri nets. The work of [Degano *et al.*, 1989] has a strong algebraic flavour. It provides axiomatizations of the various kinds of identifications one would like to make between the computations of a Petri net in an abstract setting. The strongest possible identification (in terms of the category $T[N]$) happens to coincide with the identification of concrete processes through the swap operation proposed in [Best and Devillers, 1987]. Now it is straightforward but laborious to set up a 1-1 correspondence between our traces and the equivalence classes of finite processes generated by the swap operation in [Best and Devillers, 1987]. This would then extend to the ("equivalence classes of") arrows in $T[N]$. We have not done so here because the scope and aim of our paper and of the two papers [Degano *et al.*, 1989; Best and Devillers, 1987] are very different from each other. We are mainly interested in fitting together the trace behaviour of a Petri net into a single object. Of course this idea of trace behaviour can be stated within the framework of [Degano *et al.*, 1989], but it will not be particularly profitable to do so. In the case of [Best and Devillers, 1987], since the equivalence classes of processes do not have a "net" representation, it is difficult to imagine how they can be sewed up together to form a coherent whole. In [Vogler, 1990] an attempt is made to find such a representation, but it seems difficult to work with this representation (which is no longer an acyclic object) except for extremely restricted subclasses of Petri nets, for instance $S$-nets. However, we

expect the results of [Degano *et al.*, 1989] to be very relevant when one starts to investigate operations over our trace languages.

Actually, the results in this paper can be lifted to the functorial level. Consider the category with objects trace languages satisfying (PN1) through (PN4) and with arrows trace morphisms, and consider the category with objects Petri nets and with arrows the strengthened version of Winskel's morphisms [Winskel, 1987b] used in [Mukund, 1992]. Then the construction from Petri nets to trace languages given in Definition 2.9 and the construction from trace languages to Petri nets given in Definition 3.7 can be extended to obtain functors between these categories. These functors form a co-reflection with the functor from trace languages to Petri nets as the left adjoint. The details will appear in [Hoogers, 1994].

Finally, we would like to return briefly to the main theme of the paper, which is to understand the behaviour of general Petri nets. In particular, we would like to associate a single behavioural object with each Petri net which would simultaneously capture the non-sequential and branching aspects of the behaviour of a Petri net. Here we have proposed a poset of general traces of a certain kind as this behavioural object. One of the motivations underlying such a "trace" approach is that it might lead to a corresponding notion of event structures, as is certainly the case for 1-safe Petri nets. (As pointed out earlier, getting at an event structure via the unfolding route does not seem to be a viable option for Petri nets.) We are glad to conclude by mentioning that this hope of getting at a proper notion of event structures suitable for describing the behaviour of Petri nets by studying the trace behaviour of Petri nets does not seem to be entirely misplaced [Hoogers *et al.*, 1993].

## ACKNOWLEDGMENTS

The authors thank the referees for detailed comments which have helped to improve the presentation of the paper. The authors gratefully acknowledge support from the ESPRIT basic research Action 3148, DEMON, and the Dutch National Concurrency Project REX sponsored by NFI.

Received March 6, 1992; final manuscript received April 4, 1994

## REFERENCES

Aalbersberg, IJ. J., and Rozenberg, G. (1988), Theory of traces, *Theoret. Comput. Sci.* **60**, 1–82.

Arnold, A. (1991), An extension of the notions of traces and of asynchronous automata, *RAIRO Inform. Théor. Appl.* **4**, 355–393.

Best, E., and Devillers, R. (1987), Sequential and concurrent behaviour in Petri net theory, *Theoret. Comput. Sci.* **55**, 87–136.

Best, E., and Fernandez, C. C. (1988), "Nonsequential Processes," EATCS Monographs on Theoretical Computer Science, Vol. 13, Springer-Verlag, Berlin/New York.

Degano, P., Meseguer, J., and Montanari, U. (1989), Axiomatizing net computations and processes, *in* "Proceedings, LICS 1989," pp. 175–185.

Ehrenfeucht, A., and Rozenberg, G. (1990), Partial 2-structures. II. State spaces of concurrent systems, *Acta Informat.* **27**, 348–368.

Engelfriet, J. (1991), Branching processes of Petri nets, *Acta Informat.* **28**, 575–591.

Goltz, U., and Mycroft, A. (1984), On the relationship of CCS and Petri nets, *in* "Lecture Notes in Computer Science," Vol. 172, pp. 196–208, Springer-Verlag, Berlin/New York.

Goltz, U., and Reisig, W. (1983), The non-sequential behaviour of Petri nets, *Inform. and Control* **57**, 125–147.

Goltz, U., and Reisig, W. (1985), CSP-programs as nets with individual tokens, *in* "Lecture Notes in Computer Science," Vol. 188, pp. 169–196.

Hoogers, P. W. (1994), "Behavioural Aspects of Petri Nets," Ph.D. Thesis, Leiden University.

Hoogers, P. W., Kleijn, H. C. M., and Thiagarajan, P. S. (1993), Local event structures and Petri nets, *in* "Lecture Notes in Computer Science," Vol. 715, pp. 462–476, Springer-Verlag, Berlin/New York.

Mazurkiewicz, A. (1987), Trace theory, *in* "Lecture Notes in Computer Science," Vol. 255, pp. 279–324, Springer-Verlag, Berlin/New York.

Mazurkiewicz, A. (1989), Concurrency, modularity, and synchronization, *in* "Lecture Notes in Computer Science," Vol. 379, pp. 577–598, Springer-Verlag, Berlin/New York.

Mukund, M. (1992), Petri nets and step transition systems, *Int. J. Foundations Comput. Sci.* **3**, 443–478.

Nielsen, M., Plotkin, G., and Winskel, G. (1981), Petri nets, event structures and domains, I, *Theoret. Comput. Sci.* **13**, 85–108.

Nielsen, M., Rozenberg, G., and Thiagarajan, P. S. (1990), Behavioural notions for elementary net systems, *Distrib. Comput.* **4**, 45–57.

Nielsen, M., Rozenberg, G., and Thiagarajan, P. S. (1992), Elementary transition systems, *Theoret. Comput. Sci.* **96**, 3–33.

Nielsen, M., Rozenberg, G., and Thiagarajan, P. S. (to appear), Transition systems, event structures and unfoldings, *Inform. and Comput.*

Rozenberg, G., and Thiagarajan, P. S. (1986), Petri nets: Basic notions, structure and behaviour, *in* "Lecture Notes in Computer Science," Vol. 224, pp. 585–668, Springer-Verlag, Berlin/New York.

Rozoy, B. (1990), On distributed languages and models for distributed computation, *in* "Lecture Notes in Computer Science," Vol. 469, pp. 434–456, Springer-Verlag, Berlin/New York.

Vogler, W. (1990), Representation of a swapping class by one net, *in* "Lecture Notes in Computer Science," Vol. 424, pp. 467–486, Springer-Verlag, Berlin/New York.

Vogler, W. (1991), A generalization of traces, *RAIRO Inform. Théor. Appl.* **2**, 147–156.

Winskel, G. (1987a), Event structures, *in* "Lecture Notes in Computer Science," Vol. 255, pp. 325–392, Springer-Verlag, Berlin/New York.

Winskel, G. (1987b), Petri nets, algebras, morphisms, and compositionality, *Inform. and Comput.* **72**, 197–238.